Design and implementation of fast and sparsified readout for Monolithic Active Pixel Sensors

Olav Torheim

Thesis for the degree of Philosophiae Doctor (PhD)
at the University of Bergen

2010

Contents

Acknowledgments **v**

Summary **vii**

1 Solid state pixel detectors for high energy physics **1**
- 1.1 Operation principle of silicon detector 2
- 1.2 Types of silicon detectors 3
  - 1.2.1 Sensors of high-resistivity processes 3
  - 1.2.2 Sensors manufactured in low-resistivity processes 5
- 1.3 Pixel detectors for CBM 14
  - 1.3.1 Requirements 15
  - 1.3.2 Limitations and advantages of standard pixel architectures . 15
  - 1.3.3 Proposed solutions 16

2 Read out architectures for particle trajectory detectors **21**
- 2.1 Physical model - clusters 22
- 2.2 Readout and data sparsification architectures 24
  - 2.2.1 Centralized readout with rolling shutter 25
  - 2.2.2 Data driven and triggered readout with timestamping 27
  - 2.2.3 Data driven and delayed readout with timestamping 28
  - 2.2.4 Architectures suitable for MAPS 29
- 2.3 Data reduction techniques 30
  - 2.3.1 State encoding 30
  - 2.3.2 Clusterization 33
  - 2.3.3 Compression efficiencies in standard MAPS with moderate occupancies 36
  - 2.3.4 Compression efficiencies with respect to multiplicity and matrix width 38
  - 2.3.5 Comparison of clusterization and state encoding with general compression techniques 39

2.4 Segmentation of pixel matrix into sparse scan banks 40
2.4.1 Necessity of segmentation 40
2.4.2 Calculating the width of the segments 41
2.5 Line address suppression and other data formatting issues 42
2.5.1 Line address suppression 42
2.5.2 Other data formatting issues 43
2.6 On-chip buffering and memory issues 44
2.7 Conclusion 45

3 MIMOSA26 - Sensor with data sparsification 47
3.1 Achieved performances of MIMOSA series of pixel detectors 48
3.1.1 Background 48
3.1.2 The MIMOSA22 and SUZE prototypes 49
3.2 Architecture of MIMOSA26 49
3.2.1 Sensors part 51
3.2.2 Digital treatment part 54
3.3 Simulation 68
3.3.1 Simulation environment 68
3.3.2 Checking the datapath 69
3.3.3 Checking the entire design 70
3.3.4 Input patterns 74
3.3.5 Pattern reconstruction 78
3.3.6 Pertinency and weaknesses of simulation 79
3.4 Hardware tests 80
3.4.1 Test environment 80
3.4.2 Performed tests 81
3.5 Architectural improvements with respect to SUZE 83
3.5.1 Parallelization of operations 84
3.5.2 Improved clocking scheme 84
3.6 Conclusion 86
3.6.1 Achieved performances 86
3.6.2 Perspectives of improved timing 86
3.7 Perspectives for STAR and CBM 91
3.7.1 STAR: Twice as large matrix and higher data transmission rate (Ultimate-1, Ultimate-2) 91
3.7.2 CBM: Twice as high frame readout rate (MIMOSIS-1) . . . 92

4 3D ASICs for pixel detectors 93
4.1 Motivation for 3D integration in high energy physics 94
4.1.1 Requirements of future pixel detectors 94
4.1.2 Technological features offered by 3D integration 94
4.1.3 3D integrated architectures for providing fast, intelligent small-pitch pixels 96
4.2 Types of 3D integration techniques 98
4.3 3D integration suppliers 100
4.4 From hybrid detectors to 3D 101
4.5 From MAPS to 3D - Internal IPHC prototypes 103
4.5.1 3-tier architecture: Strip-like 104
4.5.2 3-tier architecture: Rolling shutter with discriminator in every pixel 106
4.5.3 2-tier architecture: Timestamping and delayed readout . . . 107
4.6 Conclusion 121

5 Conceptual design for a CBM 3D integrated detector 123
5.1 Requirements for a 3D integrated pixel detector for MVD 126
5.1.1 Spatial and time resolution 126
5.1.2 Radiation hardness 127
5.1.3 Power consumption 128
5.1.4 Data reduction 129
5.1.5 Proposed architectures 130
5.2 Data driven readout with tokens and distributed state encoding . . 133
5.2.1 Distributed state encoding and token passing between pixels in each line 133
5.2.2 Line segment structure 137
5.2.3 Readout 142
5.2.4 Floorplanning and layout 146
5.2.5 Serial output links 146
5.2.6 Noisy pixel contribution to data stream 150
5.2.7 Power consumption 151
5.2.8 Scalability of architecture 153
5.3 SUZE on top of the rolling shutter 157
5.3.1 Dimensioning SDS and multiplexer 158
5.3.2 Dimensioning the memory 159
5.3.3 Floorplanning and layout (area estimations) 159
5.3.4 Serial output links 160
5.3.5 Power consumption 160

5.3.6 Scalability of architecture 161
5.4 Conclusion with comparison of solutions 163

6 Summary and outlook 165

A Publications 169

B Comparison of clusterization and state encoding with JPEG 171

C Control registers for MIMOSA26 175

D Test patterns for memory and datapath of MIMOSA26 177
D.1 Memory test structures 177
D.2 Two-line patterns for propagating failure in the datapath 178

E ILC perspectives for MIMOSA26 181
E.1 ILC: Relaxed data transmission between bunch trains 181
E.1.1 Redimensioning the multiplexer or replacing state encoding with clusterization 181
E.1.2 Replacing continuous data transmission with delayed data transmission 182
E.2 ILC outer layers: Lower power and higher pixel accuracy 184
E.2.1 Requirements derived from physics experiment 184
E.2.2 Data sparsification with state encoding 185
E.2.3 Data sparsification with clusterization 186
E.2.4 Implementation and memory issues with state encoding . . . 188
E.2.5 Implementation and memory issues with clusterization . . . 188
E.2.6 Memory storage considerations 189
E.2.7 Conclusion 191

F Block schematics and implementation details for CBM 3D readout 193
F.1 Pixel cells 193
F.1.1 Redundant case, only seeing hits of nearest neighbors 194
F.1.2 Nonredundant case, propagating hit information as long as necessary 195
F.2 Line buffers 198

G Abbreviations 201

Acknowledgments

This work has been carried out within the Microelectronics Research Group at the Department of Physics and Technology, University of Bergen, and the IPHC at CNRS in Strasbourg, during the period of January 2008 to May 2010.

At the University of Bergen, I would like to thank my supervisor Kjetil Ullaland and my co-supervisor Dieter Röhrich for guidance and support throughout the period of work. Shiming Yang for helping me with the part of my thesis related to CBM conceptual design. I would also like to thank Kjell Brønstad, Arne Solberg, Nikolai Østgaard and Yngve Skogseide for good cooperation during the first year of my PhD scholarship when I was working with the SIR-2 spectrometer.

At IPHC in Strasbourg, I would like to thank Marc Winter and Christine Hu for providing me with interesting and challenging work. I would also like to thank Guy Doziere and Andrea Brogna for helping me with my work and with writing my thesis. Similarly I would also like to thank Abdelkader Himmi, Xiachao Fang, Yunan Fu, Andrej Dorokhov, Wojciech Dulinski, Claude Colledani, Rita De Masi, Jerome Baudot, Matthieu Goffe, and Yavuz Degerli (IRFU-Saclay). And I would like to thank to Michael Deveaux for guiding me in the right direction during my work with conceptual design for CBM and providing me with the simulation results of the CBM MVD.

Then I thank my friends Alexandre Thöne, Elodie Vinandy, Alexandre Junger, Emmanuel Rapp, and Yann Heintz, for motivating me to learn french and taking me to a lot of exciting adventures throughout France and Europe.

Finally, I would also like to thank my parents Henny and Per Torheim for their support. Also my daughter Marie Torheim for providing me with inspiration for finishing my studies as quickly as possible.

Olav Torheim, Bergen, September 2010.

Summary

This thesis focuses on the development of smart pixel readout architectures that should ultimately be targeted for the Micro-Vertex Detector (MVD) of the CBM (Compressed Baryonic Matter) experiment. The technical challenge of building a pixel detector for this experiment is to design particle sensors capable of meeting at the same time very strict requirements on both spatial resolution, time resolution and radiation hardness.

The MVD is required to obtain data for the open charm physics programme of CBM. For a collision rate of 10^6 collisions/second, it can be shown that the required time resolution for the first MVD detector station, 5 cm away from the target, is around 2 µs, while the required spatial resolution becomes 5 µm (assuming a material budget of 300 µm Si). The detector has to withstand a non-ionizing radiation of $2 \cdot 10^{14} neq/cm^2$ and the material budget is about 0.2-0.3 % of radiation length.

While bump-bonded hybrid detectors meet the requirements on time resolution and radiation hardness, they do not provide the required spatial resolution and material budget. Detectors based on Monolithic Active Pixel Sensors (MAPS), on the other side, have sufficiently high spatial resolution and low material budget, but they do not meet the requirements on time resolution and radiation hardness. To obtain a detector that has the superior spatial resolution and material budget of MAPS while at the same time providing the fast readout rate and high radiation tolerance of hybrid detectors, new technologies must therefore be explored.

Increasing time resolution also means increasing the amount of raw data to transfer. With 2 µs frame readout time, and a typical pixel matrix size of 500,000 binary discriminated pixels, the raw data rate of each sensor becomes 250 Gbits/sec. To keep the amount of serial data links from the sensors at an acceptable level with respect to bonding and material budget, and to avoid overloading the data acquisition system, it is also necessary to embed data reduction or data compression functionalities within the sensor readout electronics.

Towards fulfilling the requirements of the needed intelligent pixel detectors, numerous steps have been taken. At the IPHC laboratories of CNRS in Strasbourg,

work has been done and is under progress to develop such detectors. The work is done step by step, starting with the most primitive and low bandwidth CMOS pixel sensors while making progress towards ever more complex and sophisticated detectors, with techniques like zero suppression and 3D integration - all techniques that will be explained in detail later in this thesis.

In the first generation of IPHC MAPS based pixel sensors, the individual pixels were externally addressed and the pixel matrix was read out entirely [1]. This means that even for detecting a few hits, the entire matrix had to be read out. In fact, as the pixel signal is at the same order as natural process variations, the pixel matrix had to be read twice and the hit signals extracted through offline correlated double sampling. In the second generation of chips, the frame readout time was reduced by embedding correlated double sampling and row-wise parallel readout, called rolling shutter operation. However, without any kind of data compression, such a reduction in the frame readout time comes at the expense of an ever increasing data rate. My contribution to the development of the MIMOSA26 sensor and the first 3D integrated detectors based on MAPS technology will be discussed in the following:

Starting my work at IPHC in march 2008, the first steps had been taken towards embedding data reduction micro circuits in the periphery of the active pixel matrix. A prototype circuit, SUZE, interfacing two programmable pixel rows of 128 columns, had been designed, manufactured, and tested, proving the principle of zero suppression. The next step, where I did my first contribution, was then to build the zero suppression into an actual sensor, the MIMOSA26. As MIMOSA26 had 1152 columns, nine times more as those interfaced by SUZE, one had to completely redesign core parts of SUZE for remaining at an operating frequency of 100 MHz.

MIMOSA26 was developed and manufactured for providing fast and sparsified readout for the EUDET-JRA1 beam telescope [2]. Although being a large step forward in terms of reduced frame readout time (100 μs), the row-by-row processing of the entire matrix becomes a bottleneck for further reduction in frame readout time. With dual-sided readout and smaller feature size, the architectures based on MIMOSA26 are expected to provide frame readout times down to 25 μs. However, at some point it becomes difficult to decrease the frame readout time due to the inherited architectural limitations. The next step towards improving the time resolution of the monolithic active pixel detectors is therefore to develop a new architecture that utilizes the new degrees of freedom offered by 3D integration.

With the first 3D detector prototypes developed at IPHC, two architectural approaches were followed: The first one, targeted at the inner layers of the International Linear Collider (ILC), was to power on all the pixels and provide the analog pixels with a digital tier containing timestamp circuitry ($\frac{950\,\mu s}{2^5} \approx 30\,\mu s$ time resolution) and to read out the timestamp information between the bunch train. The analog pixels could then be turned off completely during the dead time to save power. To provide this architecture with a faster data transmission using a minimum of bonded wires, a readout control circuit with 8b10b serial transmission was designed and implemented.

The other approach, required for continuous beam experiments like CBM, continues with rolling shutter operation, but the time for processing each row is reduced by taking advantage of 3D technology to embed a discriminator into each pixel. As the bottleneck of rolling shutter operation still is the fixed line processing time, the next measure to reduce frame readout time is parallellization by splitting into rolling shutter segments operated in parallel.

To arrive at a frame readout time as low as 2 µs, required for CBM, and two orders of magnitude lower than the so far achieved frame readout times, the recently submitted 3D integrated rolling shutter architecture was chosen as a starting point for continuing the 3D detector development in the direction necessary for this experiment. Already interfacing a binary discriminated rolling shutter circuit with excellent line processing time, an important part of this work would then be to prevent the digital readout from lagging behind this improved rolling shutter circuitry, and similar measures of parallellization were therefore proposed for the digital zero-suppression circuitry.

Instead of keeping the zero-suppression circuitry in the periphery, where it has to process lines one by one at the same speed as that of the rolling shutter, a new and pixelized structure is proposed, where the zero suppression is distributed to the individual pixels, and with the pixel rows extracting their hit information in parallel. To combine the sequential rolling shutter operation of a submatrix with a parallel search for hit patterns, each submatrix splits into two halves, where the rolling shutter injects hits in one half while the zero suppression circuitry extracts pattern information in the other half.

The proposed design has been verified through simulation. Compared to the zero suppression circuitry of MIMOSA26, the proposed new readout is superior with respect to scaling with a higher hit density, and it does not have the limitations of a maximum number of hits to extract from each row, only a maximum number of hits to extract from the entire matrix.

The thesis is organized into six chapters, with the last chapter containing a summary and an outlook:

Chapter 1 starts with a general description of different solid state detectors and their integration with electronics, like hybrid and monolithic, followed by a description of the MAPS architectures and the strategy chosen to provide these sensors with faster readout.

Chapter 2 gives a discussion on the state of the art of readout architectures for pixel detectors. It also gives a discussion on the readout electronics of the latest MAPS-based sensors and their requirements, especially focusing on the concepts of zero suppression and clusterisation.

Chapter 3 presents the MIMOSA26 pixel architecture, with its fast line processing and sparsified readout. It also presents the work that has been done with creating a test environment for verifying and validating the MIMOSA26 design prior to submission, and it presents the future perspectives of adapting this architecture to new experiments.

While chapter 3 presents the last achievements in 2D MAPS, chapter 4 introduces the new techniques of 3D integration. Vertical 3D integration offers the possibilities of using the best features from different processes, and to increase the amount of logic per pixel for providing functionalities previously available only in hybrid detectors. With 3D integration, it is also possible to split rolling-shutter operated MAPS into smaller segments that are operated in parallel, thus avoiding the rolling shutter from becoming a bottleneck for the time resolution.

The same chapter also introduces the three first 3D chips designed at IPHC. The first two chips are 3-tier designs with data driven readout or rolling shutter operated readout, respectively. The third chip, which is described in more detail, is a two-tier prototype chip, targeted for ILC, which is implemented with time stamping and delayed readout, and with an 8b10b readout interface for fast serial transmission.

In chapter 5, a conceptual design is presented for a 3D integrated detector that meets the requirements of CBM. In the proposed design, the zero-suppression techniques of the rolling-shutter based 2D MAPS is combined with the new features of the first 3D integrated circuits. Rolling-shutter operation is still utilized to save power, but the matrix is split into segments to meet the requirements of time resolution. The zero-suppression circuitry that was placed in the periphery of the 2D matrices has been distributed into the pixels. By splitting each submatrix at the digital tier into two halves that are read out at each their time interval, the sequential rolling-shutter operation of rows in the analog tier is combined with parallel token-injection in the rows of the digital tier.

Through segmentation and parallellization, the performance of the readout electronics is raised to a level where it is possible to arrive at 2 μs frame readout time and thus meet the time resolution requirements of CBM.

Chapter 1

Solid state pixel detectors for high energy physics

High energy particle accelerators generate particle collisions at interaction rates up to 40 MHz, with around 1000 new particles coming out from each collision (Large Hadron Collider at CERN). As some of the rarest particles do not live more than a picosecond or less, they have to be indirectly identified by reconstructing their secondary vertices, indicated by the trajectories of their daughter particles. To be able to separate the daughter particles of the decay vertex from the background particles of the original collision vertex, it is necessary to detect these particles as close to the interaction point as possible, and with the highest possible spatial accuracy [3].

A pixel detector is composed of a planar two-dimensional array of particle sensing elements, where each individual element is called a pixel. Future high-energy projects, like the Compressed Baryonic Matter (CBM) experiment at FAIR and the International Linear Collider (ILC), have led to lots of efforts being put into developing solid state pixel detectors for particle trajectory reconstruction [4] [5] [6].

With CBM and ILC, it is intended to use pixel detectors for providing detection and identification of short lived hadrons, like for example D mesons. As such hadrons decay into daughter particles like kaons and pions long before they even reach the first detector element, they must be identified by reconstructing their decay vertices.

Although the daughter particles may travel through long distances during their lifetime, every interaction with matter results in scattering and thus a deviation from the original path. To be able to measure the original particle trajectories, the pixel sensors must therefore be located in the innermost layer of the detector, and with a minimum of material budget for cooling and mechanical support [1]. And due to the very small radius of the innermost layer, these sensors must also be of very high spatial resolution.

1.1 Operation principle of silicon detector

The silicon detector is a solid state crystal operated as a reverse biased p-n diode. The p-n diode is created by implanting acceptors and donors in the material, like boron and phosphorous. When the diode is reverse biased, only a small leakage current flows through the depletion zone in the p-n junction, but if an ionising particle traverses the sensor, it ionizes the medium and excites electrons from the valence band to the conduction band. Electron-hole-paires are thus created at a typical rate of 80 electron/hole pairs per micrometer, and electrons and holes are swept to each their side by the electrical field of the p-n junction.

A block schematic of a general particle detection system is shown in Figure 1.1. A reverse-biased photosensitive p-n diode detects charged particles as electron-hole pairs are generated by ionizing particles inside the detector material. The generated electrons drift in the electric field towards the positively biased n-electrode of the detector. The total induced charge, Q, is integrated onto the feedback capacitor of a charge sensitive amplifier, thus giving rise to a voltage step of amplitude Q/C_f at the amplifier output.

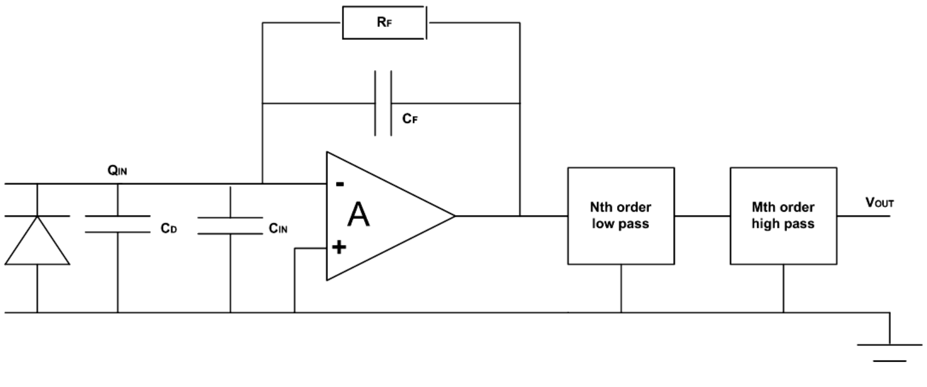


Figure 1.1: Charge Sensitive Amplifier (CSA) with shaper. The detector sensing element is modeled as a reverse biased p-n diode with a parasitic capacitance C_D, while the shaper is modeled as a low pass filter followed by a high pass filter (RC-CR).

[1] In detector development, material budget is a constraint on the detector thickness that is specified in percentage of radiation length.

The step signal from the charge sensitive amplifier is fed to a pulse shaping amplifier. The main purpose of pulse shaping is to optimize the signal-to-noise ratio (SNR) of the readout electronics and to provide a signal that is suitable for further signal processing. The shaper is especially beneficial for reducing the contribution of high and low frequency noise coming from the preamplifier and the detector. The shaper can be represented as a series connection of N first order low pass filters and M first order high pass filter.

Depending on the type of application, the stage following the pulse shaper could be a discriminator for event detection or a peak detect and hold circuit for pulse height measurement [7] - or a combination of both features, as in the n-XYTER chip that was used for high rate imaging neutron detection [8].

It can be shown that for a typical detector setup with charge sensitive amplifier and shaper, the equivalent noise charge increases more or less linearly with the detector capacitance and the preamplifier input capacitance [7]. The smallest possible detector capacitance and preamplifier input capacitance is therefore desired.

1.2 Types of silicon detectors

Although solid state detectors have been known for more than 50 years, it was first in the beginning of the 1980s, with the search for short-lived particles in experimental physics, that their particle tracking capabilities were investigated.

The introduction of planar technology gave first rise to the one-dimensional devices, the strip detectors. The next devices in line were the two-dimensional planar devices, like double-sided strip detectors, and charge coupled devices (CCD). Another essential step came with the development of hybrid devices, where pixelized planar devices could be bump-bonded to pixelized readout ASICs [9], and the development of monolithic devices, where sensor and readout electronics are on the same device.

1.2.1 Sensors of high-resistivity processes

Microstrips

With the strip detector, narrow strips of electrodes are implanted in parallell on a silicon substrate of high-purity bulk material. This configuration results in several long and narrow p-n diodes with the same properties as a single diode. By simply combining perpendicularly oriented strip detectors, or by implementing perpendicularly oriented strips on the sensor backside, a point-like p-n diode appears at every strip intersection. In such case we get a two-dimensional position sensitive detector [10].

The strips of the silicon strip sensors are normally wire-bonded to their readout electronics. Example of such a sensor is the SCT microstrip sensor used in ATLAS [11]. In the research and development phase of the ATLAS detector, different kinds of options for strip detectors were prototyped and evaluated with respect to constraints like radiation hardness and manufacturability. Double-sided sensors were prototyped, and also sensors with n^+ electrodes in an n-type substrate, but they were rejected at an early stage in favour of single-sided sensors with n^+ electrodes in a p-doped substrate (largely because of system difficulties with readout at high voltage) [12].

Planar pixel sensors

The planar pixel sensor is a planar array of p-n or p-i-n diodes, whose electrodes are coupled to their pixelized readout electronics through bump-bonding. Their accompanying readout chips bring separate logic to every pixel cell, and such detectors are called hybrid detectors.

Compared to microstrip detectors, the segmentation of the electrodes reduces the capacitance and the leakage current per channel, thus improving the SNR. An example of such a sensor is the pixel sensor used in the innermost layer of the ATLAS particle tracking system [13].

The pixel sensors used in ATLAS collect signals on n^+-electrodes - the n-in-n sensor, where the p-n junction is fabricated on the backside [14]. As the type of silicon wafer inverts into p-type after irradiation, the n-in-n sensor becomes an n-in-p sensor, where the p-n junction is moved to the front side. An advantage of the n-in-p sensor is thus the possibility of continuing to operate the sensor in partial depletion prior to irradiation. The charge collection efficiency is also enhanced by collecting electrons in the n^+-electrode, instead of holes in a p^+-electrode, which are more prone to trapping due to their lower mobility.

DEPFET

The detector concept of DEPFET (DEpleted P-channel Field Effect Transistor) was developed in the late eighties [15]. The principle of DEPFET is to use an n^+ doping implant - and appropriate biasing of bulk, source, drain and a p^+ rear electrode - to establish a local potential minimum right underneath a p-channel field effect transistor. While holes created in the bulk material are drifting towards the rear electrode, electrons are collected in the local potential minimum underneath the channel. The potential minimum will act as an internal gate for the transistor, and the electrons collected are thereby modulating the transistor current.

Due to small internal gate capacitance, absence of external connections to the first amplification stage, and a fully depleted bulk, an excellent SNR can be obtained with DEPFET devices. Pixel detectors based on DEPFET are to be installed in the vertex detector of the BELLE-2 experiment [16] and are also among the candidates for equipping the vertex detectors of ILC [17][18].

3D

In the late nineties, a new architecture for silicon detectors was proposed, called the 3D detector [19]. In this detector, the electrodes are penetrating through the entire substrate. Since the width of the depletion region depends on the spacing between the electrodes, and not any more on the wafer thickness, it is possible to obtain low depletion voltage and fast charge collection times.

In heavily irradiated planar detectors, the increase of the depletion voltage and the reduction of the charge carrier mean drift length due to trapping effects are a major issue. In 3D detectors, a direct consequence of the low depletion voltage and short distance between electrodes is an increase of the radiation tolerance [20].

Unlike other detectors, which have a substantial dead area along their edges, 3D detectors have active edges, which are formed from electrodes in the third dimension, perpendicular to the top and bottom surfaces, with full sensitivity to within a micron of the physical border [21].

3D sensors with electrodes penetrating the entire substrate and with active edge were successfully tested with the ATLAS pixel readout electronics. This technology gives the possibility of constructing sensors with depletion voltages an order of magnitude lower than the depletion voltages of their corresponding planar sensors, and sensitivity to within a few µm of the physical edges. A drawback of 3D detectors is, however, the rather long and complex fabrication process, as methods of silicon micro-machining must be used. This could lead to the unfeasibility of mass production of these detectors [20]. Another problem is that the 3D electrodes are parallell to the particle tracks at normal incidence, and this leads to considerable signal loss in the cases where particles pass through the inside of electrodes, reducing the detection efficiency [22].

1.2.2 Sensors manufactured in low-resistivity processes

Since both detector and readout electronics can be based on silicon, it is in principle possible to build sensor and electronics into the same wafer, with readout electronics on the surface and the bulk material as the sensitive volume. Without wire bonding or bump bonding between sensor and electronics wafers, the total input capacitance is strongly reduced, and in this way excellent noise figures obtained - while at the same time avoiding the costly hybridization processes. And when no more limited by the minimum dimensions needed for bump bonding (see for example [23] and [24]), much smaller pixels can also be implemented.

However, while commercial CMOS processes for electronics need low resistivity substrates to fabricate small transistors, silicon detectors need high-resistivity substrate for depleting their active volume. A monolithic pixel detector with non standard CMOS on high-resistivity bulk material was successfully prototyped already in the early nineties [25], but no large scale detectors have been developed on this approach, due to the use of nonstandard processing technologies.

If standard CMOS processes are to be used for manufacturing monolithic pixel detectors, there are currently two different options. The first one is CMOS with charge collection in the epitaxial layer[2] (MAPS), and the second one is silicon on insulator (SOI), where the support wafer underneath the isolating oxide serves as detector layer for the amplification and readout electronics located on top.

The principle of using the epitaxial layer as active volume is the following:

Since the impurity level of the epi layer - and therefore also its resistivity - is lower than the impurity level of the bulk material, alignment of the different fermi levels in the two layers lead to a potential barrier being created at the boundary between the epi and the bulk material. When electron-hole paires are generated by impinging particles in the epitaxial layer, the generated holes will therefore quickly disappear into the lower-potential substrate, while the generated electrons remain within the higher-potential epitaxial volume.

In pixel detectors based on CCD, the epitaxial volume is patterned into potential wells where the electrons are stored until being transferred out of the pixel matrix with a three-phase clock. In pixel detectors based on MAPS, the electrons diffuse thermally around until reaching the charge collecting diode of the pixel.

Charge coupled devices (CCD)

The operating principle of charge coupled devices is to create an array of gates where potential wells for trapping electrons are formed underneath them. A basic CCD cell is comprised of a photogate to provide the potential well and two isolation gates to provide isolation to the neighbouring wells. When the content of the potential

[2]An epitaxial layer is a thin single crystal layer grown on top of an underlying crystal, with the latter providing the desired lattice structure. A variety of methods can be used to provide the appropriate atoms to the surface of the growing layer, like liquid-phase epitaxy or molecular beam epitaxy.

wells is to be read out, a three-phase-clock is used to shift the potential wells from one cell to another. With this behavior, the rows of accumulated charges may be moved one by one to the periphery of the chip, where they are converted to voltages by a sense amplifier.

The active area of a charge coupled device is in the epitaxial layer of the chip, normally of around 20 µm depth. Since most of the chip volume does not contribute to the electrical signal, CCD circuits for particle trajectory detection can be thinned down without losing signal to meet the requirements of material budget [26].

A main disadvantage of using CCD circuits in particle physics experiments is their long readout times. The charge has to be shifted serially under one gate to the next one, across the rows and columns of pixels, and due to the slow three phase clock and the large number of rows, the readout time of the whole detector may reach hundreds of milliseconds. For example, with the VXD3 CCD chips installed in the SLD detector, the readout time was 200 ms [27]. However, as part of the R&D for ILC, CCD prototype chips aiming at a time resolution as low as 50 µs have been developed. The prototypes have been based on two parallell approaches. With the first approach, sequential readout has been replaced with column parallell readout. With the second approach, each pixel cell is equipped with 20 different charge storage elements, with each element collecting the charge deposited under the pixel photogate at each their 50 µs time interval - followed by delayed readout between the 1 ms bunch trains [28].

Poor radiation hardness is another weak point of CCD devices, as charge trapping occurs in numerous bulk transfers, resulting in the deterioriation of the charge transfer efficiency. Even very small charge transfer inefficiency can lead to significant losses and signal smearing, and the possible number of rows of a CCD matrix is thus limited due to post-irradiation effects [29].

Monolithic Active Pixel Sensors (MAPS)

With monolithic active pixel sensors, the generated electrons released by an ionising particle are kept in the epitaxial layer by potential wells at the boundary until they reach an n-well collection diode by thermal diffusion. Due to the thin active volume, with typical epitaxial depths from 10 to 20 µm, the total signal charge of MAPS is very small ($< 1000e^-$) compared to fully depleted high-resistivity devices, and the charge collection time is in the order of 100 ns due to the slow thermal diffusion transport in the epi potential well. Development of very low noise electronics is therefore required for this kind of detector.

The operating principle of a monolithic active pixel sensor (MAPS) is shown in Figure 1.2. The n-well acts as charge collecting diode, while the p-wells contain the transistors required for preamplification. In cross section B, there is a positive contact potential difference on each side that prevents the generated charge from escaping the epi volume. In cross section B, the lightly doped p-epi sees two negative contact potentials, one on the bottom and another one on top. The negative contact potentials are due to the much larger doping concentration of both the p++ substrate on bottom and the p-well on top. Electrons diffusing around in the p-epi are therefore prevented from escaping the epi volume. In cross section A, the p-epi sees a positive contact potential on top and a negative contact potential on bottom. The positive contact potential is due to the p-n junction between the epi and the n-well, and eletrons diffusing into this depletion region are swept across the electrical field and collected by the n-well.

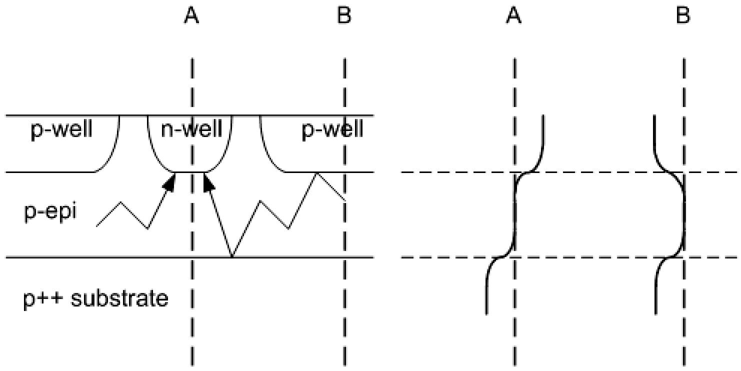


Figure 1.2: MAPS operation principle. A cross sectional view showing the charge diffusion to the left, and with the corresponding potential distribution of the two cross sections A and B shown to the right.

Figure 1.3 shows the response of a MAPS structure to X-ray photons from a ^{55}Fe-source at energies of 5.9 keV. The MAPS structure is depleted only directly under the n-well collection diode where full charge collection is obtained, and the peak corresponds to photons absorbed in the small depleted region of the n-well. However, the main part of the spectrum corresponds to the absorption of photons in the undepleted part of the epitaxial layer, where the charge collection is incomplete due to recombination of the diffusing electrons and holes [1].

In principle, it is also possible for electrons generated in the bulk to diffuse to the boundary and become captured in the epitaxial potential, contributing to the signal. But since the life time of charge carriers inside a heavily doped p^{++} substrate is so short, this contribution is in practice negligible, and the active volume is thus limited to the epi layer. With an epi layer of thickness in the order of

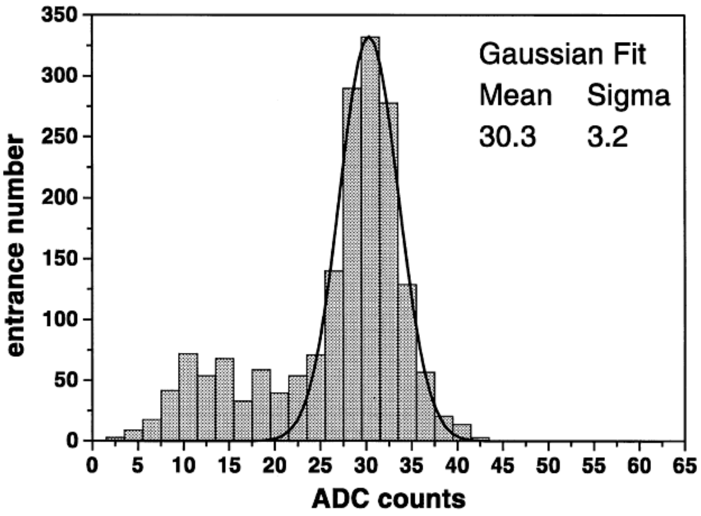


Figure 1.3: Distribution of collected charge for 5.9 keV photons [1].

10 µm one can expect to collect close to thousand electrons created by a minimum ionising particle. However, a moderately doped substrate without epitaxial layer has also been tried with excellent minimum ionizing particle tracking performance. These results open a much larger choice of possible CMOS processes in the future, which are important due to the observed tendencies of either removing or drastically reducing the epi thickness in very deep submicron technologies. Still, the absence of electrons reflected by the epitaxial/substrate barrier and the deeper origin of the substrate electrons results in a wider and less peaked charge distribution of clusters. With the MIMOSA-4, manufactured with 20 µm pitch in non-epitaxial process, a single-point resolution of approximately 4 µm was observed in the beam tests, compared to approximately 1.5 µm for its epitaxial-manufactured counterparts [30].

Normal MAPS chips cannot utilise both the n- and p-types of transistors at the same substrate, since competing n-wells required for p-type transistors would draw charges away from the charge collecting diode. All parts of electronics that require both kinds of transistors must therefore be replaced with structures that only utilise n-type transistors - or they have to be moved to the periphery of the matrix. Pixel cells that follow the traditional configuration with CSA, shaper, PDH and discriminator are therefore difficult to implement. CSA and shaper are instead replaced with a low gain amplifier and source follower, with the corresponding requirements for low noise design.

Figure 1.4 shows a block diagram of a basic MAPS pixel cell, each being composed of three transistors - a self biasing diode [3], a source follower and a pass transistor for connecting to pixel cell to the column line. As charge is collected on the diode, a voltage corresponding to $v_q = \frac{q}{C_{diode}}$, is generated, and with C_{diode} being the diode capacitance. All the amplification circuitry of the pixel cell can be switched off during charge acquisition, with the purpose of saving power. This circuitry is instead only switched on for the small period when the row of this pixel is processed for readout.

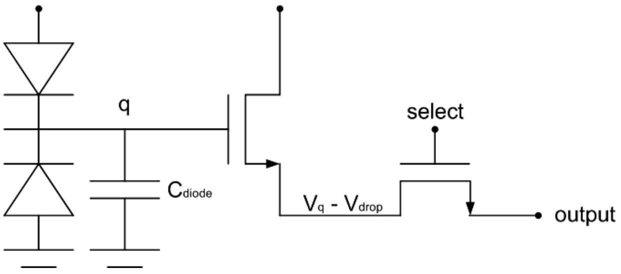


Figure 1.4: Block schematic of a basic MAPS pixel cell.

The current passing through the self biasing forward diode is very low, and hence, its $1/g_m$ impedance[4] seen from the charge collection electrode is very large. A time constant in the order of seconds before returning to the DC operating point is therefore obtained [5].

The amplitudes of the pixel signals are in the millivolt range, which is within the voltage deviations caused by process variations. It is therefore necessary to individually correct the pixel signals for variations in offset through correlated double sampling. With correlated double sampling, the pixel is sampled twice: First, the pixel signal is sampled and stored onto a capacitor, then a global reference value is sampled onboard the pixel cell and stored on another capacitor. The potential difference between the two signals can then be A/D converted or discriminated.

At IPHC, the MIMOSA (Minimum Ionizing Particle MOS Active Pixel Sensor) series of MAPS based pixel sensors have been developed for several years. Figure 1.5 shows a block diagram of a basic MIMOSA matrix, where a column shift register

[3]An alternative configuration is to replace the self-biasing diode with a switch that is used to reset the pixel to a reference value after each read cycle.

[4]The transconductance of a transistor, g_m, is its small-signal current amplification, $\frac{d_{i_{out}}}{d_{v_{in}}}$

[5]Such a long time constant could in some cases be considered a source of possible pil-eup problems. However, as the charge signal is within the order of millivolts, there has to be a very high number of hits before the pile-up voltage exceeds the valid range

and a row shift register is used to select individual pixels for readout. The logic for controlling the readout is located in the peripheral area of the active pixel matrix.

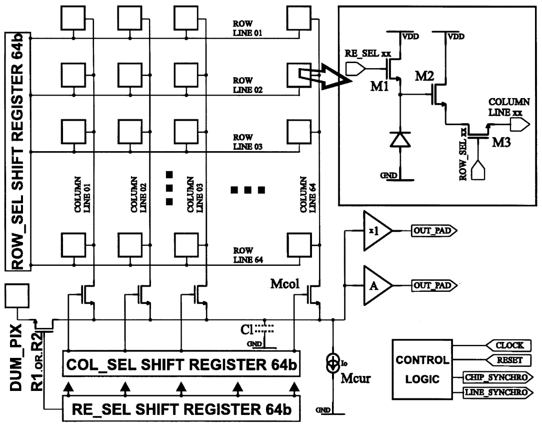


Figure 1.5: Simplified block diagram of a single array of a MIMOSA circuit [31].

As a direct consequence of one of their basic operation principles, CMOS sensors offer very high spatial resolution. Since the charges liberated when a minimum ionizing particle traverses the epitaxial layers diffuse thermally and get distributed over several pixels, the resulting charge sharing allows to reconstruct hit positions with a spatial resolution much better than the binary resolution associated with the pixel pitch (which is the pixel pitch divided by the square root of twelve).

Since only the epitaxial layer contributes to the charge collection, the pixel detectors based on MAPS can similarly to CCD be thinned down to improve material budget, without losing signal. Several MAPS-based MIMOSA chips, like MIMOSA18[32] and MIMOSA26, have been thinned down to 50 µm using commercially available post-processing techniques. Such sensors are therefore particularly suitable for low material budget operation in the vertex region of a detector.

Deep n-well MAPS

A drawback of using the MAPS detectors is the fact that only NMOS transistors can be used in the active area of the pixel matrix. Deep n-well MAPS (DNW-MAPS), which is another emerging MAPS-based technology, overcomes this problem by using a buried n-type layer as the charge collection diode. If the deep n-well electrode is made sufficiently large, smaller and shallower n-wells can be allowed in the vicinity of the larger and deeper charge collecting electrode with rather moderate, but still significant, degradation of the charge collection efficiency.

The availability of both transistors inside each individual pixel cell enables including more sophisticated functions inside the pixel cells, thus making the implementation of data sparsification and time stamping a viable solution. Examples of such designs are the SDR0 prototype chip (25 μm pitch) for the ILC vertex detector, based on timestamping and delayed readout [33], and the APSEL3D and APSEL4D prototype chips (50 μm pitch) for the KEK Super B-Factory, based on data-driven, continuous readout [34].

Although improving pixel functionality, the deep n-well designs lag behind in terms of detection efficiency. Beam test results of APSEL4D show that the detection efficiency saturates at around 92% [35], compared to detection efficiencies above 99.5% in standard MAPS chips (see for example [36]).

Depleted epitaxial MAPS

Due to demands from industry on improving the CMOS photo-detection performance, processes with resistivity high enough to fully or partly deplete the epitaxial layer have recently become available. The charge collection mechanism of MAPS sensors implemented in this technology will then be drift instead of diffusion.

The fully or partly depleted epitaxial layer is available in the XFAB-0.6 process, and a prototype chip in this process, MIMOSA25, has already demonstrated good results. The benefits of having a depleted epitaxial layer is that the released electrons will immediately drift towards their collecting electrode, instead of slowly diffusing like they do in the standard MAPS chips. Dependent on the radius of the depletion region around the charge collection diode, the charge collection time can be reduced by an order of magnitude or more, improving the radiation tolerance.

With standard MAPS, the tolerable amount of non-ionizing radiation is around 10^{12} neq/cm^2. MIMOSA25 (20 μm pitch and 15 μm deep epitaxial layer) has been tested in beam with excellent results up to $3 \cdot 10^{13}$ neq/cm^2, which has been considered promising for extending the radiation tolerance up to 10^{14} neq/cm^2 [37].

Since the electrons are directly drifting towards their nearest collection electrode, instead of diffusing in the substrate, the cluster multiplicity of the hit signal becomes much less than in standard MAPS diodes (clusters are further described in Chapter 2). This concentration of charge in fewer pixels (90% of collected charge in 3 pixels) increases the SNR, and makes it therefore possible to increase the discrimator threshold to suppress noisy pixels (especially important after irradiation) without significant loss in hit detection efficiency. Measurements from MIMOSA25 show that the collected charge in the seed pixel is at least twice larger for the sensor with depleted layer than for an undepleted one, translating into an SNR of around 60 before irradiation (and remaining above 30 after being exposed to $3 \cdot 10^{13} neq/cm^2$).

While nondepleted MAPS typically have fake hit rates below 10^{-4} for detection efficiencies above 99.5% [38], the doubled SNR of depleted MAPS a makes it possible to increase the threshold and reduce the fake hits rates by orders of magnitude while remaining at the same excellent detection efficiency.

SOI

Silicon On Insulator, SOI, is a process that has been used for many years in commercial electronics production. One of the advantages of the process is its immunity to latchup, as the individual transistors are isolated from each other, avoiding the parasitic bipolar structures.

The idea of utilizing SOI in a detector is to use the support wafer under the buried oxide as a fully depleted high-resistivity detector material. With this technique, the entire underlying substrate contributes to the charge collection, and one can therefore obtain a much larger active volume than that of MAPS, where only the thin epitaxial layer contributes to the charge collection (and with epitaxial thicknesses varying from process to process, like the 7 μm epitaxial depth of MIMOSA8, fabricated in TSMC-0.25, compared to the 14-20 μm epitaxial depth of the later chips fabricated in AMSC35 [39]).

SOI also gives the full freedom of implementing both PMOS and NMOS transistors above the buried oxide layer, with the entire support wafer functioning as a fully sensitive detector without dead area.

SOI detectors are formed by connecting a top wafer of low resistivity and a bottom wafer of high resistivity through a silicon oxide bond. Thus, a buried oxide is formed between the wafers, and after bonding the top wafer is thinned to only a few μm. Later, vias can be etched right through the buried oxide, and in this way detector and readout electronics are connected together.

Although SOI has many advantages compared to MAPS, it also introduces disadvantages like the back gate effect of the underlying support wafer [40]. Due to charge trapping in the buried oxide, which both affect the threshold voltage in the transistors, the detectors based on SOI are also highly sensitive to ionizing radiation. Analog designs in SOI technology are also difficult due to self-heating effects, poor transistor matching and poor transistor bulk contact quality compared to bulk counterparts [41].

1.3 Pixel detectors for CBM

This work focuses on the development of sparsified readout architectures for MAPS-based pixel sensors that ultimately should be targeted for the Micro Vertex Detector (MVD) of the CBM (Compressed Baryonic Matter) experiment. The technical challenge of building pixel architectures for this experiment is to design architectures capable of meeting at the same time very strict requirements on both spatial resolution, time resolution and radiation hardness.

The CBM detector is composed of the Micro Vertex Detector (MVD), a Silicon Tracking System (STS), a Ring Imaging Cherenkov detector (RICH), replaced with a Muon Chamber (MUCH) when performing muon measurements, a Transition Radiation Detector (TRD), a Time Of Flight spectrometer (TOF), and an elecomagnetic calorimeter (ECAL) [42]. Figure 1.6 shows the detector setup.

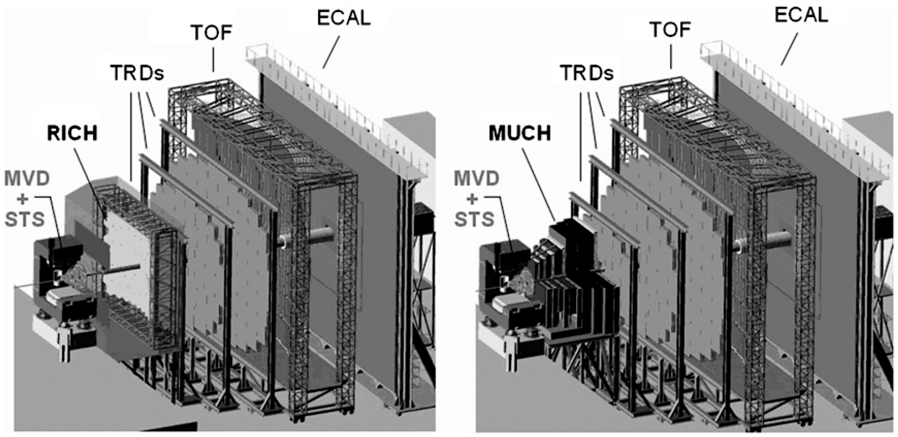


Figure 1.6: CBM detector setup. Left: Configuration for electrons. Right: Configuration for muon measurements [42].

At the energies obtained during the CBM experiment, charmed mesons are produced close to threshold. Consequently, the multiplicity of these particles is extremely small, with about $2 \cdot 10^{-4}$ D_0 mesons per central Au+Au collision at 25 AGeV. These signals must be measured via the decay into charged hadrons, and to separate these daughter particles from a background of several hundreds of other charged particles created in the primary collision, it is necessary to reconstruct the displaced decay vertices with a spatial resolution of 50 µm or better.

The heart of the CBM detector is the STS, composed of silicon strip detectors and responsible for reconstructing the particle tracks. To refine the spatial resolution of the particle tracks to the level required for the secondary vertex reconstruction of charmed mesons, the STS is preceded by two or three detector stations of MAPS-based pixel detectors, which make up the Micro Vertex Detector (MVD).

1.3.1 Requirements

Time resolution, spatial resolution, radiation hardness, and material budget

The MVD is required to collect data for the open charm physics programme of CBM, and we intend to run the experiment at 10^6 collisions/second. For such a collision rate, it can be shown that the required time resolution for the first MVD detector station, 5 cm away from the target, is 2 µs, while the required spatial resolution becomes 5 µm. The detector has to withstand a non-ionizing radiation of $2 \cdot 10^{14} neq/cm^2$ and an ionizing radiation of approximately 34 MRad, and the material budget of one detector station is just a few 0.1 % of the radiation length.

Currently there is no pixel detector that is able to meet all these requirements at the same time. To obtain a detector that has the superior spatial resolution and material budget of MAPS while at the same time providing the fast readout rate and high radiation tolerance of hybrid detectors, new technologies must be explored.

Data flow

Increasing time resolution also means increasing amounts of raw data to transfer. With 2 µs frame readout time, and a typical pixel matrix size of 500,000 binary-output pixels, the raw data rate of each sensor becomes 250 Gbit/sec. With such an enormous amount of data to read, it will, even with fast a differential series transmission protocol, be difficult to bond the sufficient amount of wires to each sensor. The large amount of wires will also make it difficult to meet the requirements of material budget.

To keep the number of serial data links from the sensors at an acceptable level, and to avoid overloading the data acquisition system, it is necessary to develop some kind of online data reduction or data compression within the sensors.

1.3.2 Limitations and advantages of standard pixel architectures

Where bump-bonded hybrid detectors provide excellent time resolution and radiation hardness, they do not provide the required spatial resolution and material budget[6]. MAPS based detectors, on the other side, have sufficient spatial resolution and material budget, but they do not meet the requirements on time resolution and radiation hardness[7].

1.3.3 Proposed solutions

Technological

For MAPS pixels, high spatial resolution and fast readout are conflicting requirement, since high spatial resolution requires small pixels while a fast readout architecture requires more logic to fit into each pixel. And as we are limited to NMOS transistors, the pixel functionalities must either be suboptimally implemented with only one type of transistors, or the pixel signal must be routed to the periphery for sequential or column parallell processing. However, 3D integration of electronics is an emerging technology providing new degrees of freedom to overcome the limitations in standard MAPS:

- Vertically integrating several wafers of pixelized chips, the detector part and the readout electronics can be separated. The limitation of NMOS-only pixels therefore no more apply. And with proper partitioning, one can design 3D stacks where different functionalities are implemented in each their most optimal technologies.
- As the available pixel area is multiplied with the number of stacked wafers, 3D integration allows pixelized readout architectures with progressively more logic per pixel, and with features similar to the readout ASICs of hybrid pixel detectors.
- Distributing the previously peripheral readout circuitry into the individual pixels, the readout electronics can be put on top of the pixel matrix, instead of being located in the periphery where it contributes to an inactive dead area.

And by replacing standard MAPS with depleted epitaxial MAPS, the charge collection time may be reduced by an order of magnitude, strongly improving the radiation hardness of the sensing element. The next step towards improving the time

[6]Like the ATLAS FE-I3 readout ASIC, with 25 ns timestamp resolution, and with modules proven to withstand $10^{15} neq/cm^2$, but with pixel pitch of 400 µm x 50 µm and interfacing a planar sensor of 250 µm thickness, [43] [13]

[7]Like the MIMOSA26, that has pixel pitch of 18.4 µm and has been thinned down to 50 µm, but with time resolution of 100 µs and a standard epitaxial substrate where charge collection efficiency is strongly degraded after $5 \cdot 10^{12} neq/cm^2$ [44]

resolution and radiation performance of monolithic pixel detectors could therefore be to develop 3D integrated pixel detectors, with depleted epitaxial MAPS serving as the sensor layer.

Architectural

In the first prototypes of the MAPS based pixel sensors, the MIMOSA chips, each individual pixel was externally addressed individually and the pixel matrix read out in its entirety [1]. This means that even for detecting a few hits, the entire matrix had to be read out, pixel by pixel. Since then, there has been an evolution in processing capabilities, where sequential processing of pixels has been replaced by parallell processing at row level (rolling shutter operation), and embedded correlated double sampling and data reduction microcircuits have been built into the sensor.

The parallell processing of rows means, however, that the frame readout time is limited by the number of rows and the time to process each row. The frame readout time can be reduced by reducing the number of rows that are processed by the rolling shutter. However, as material budget limits the number of overlapping sensor chips to two, the minimum height of the pixel matrix is equal to the height of the readout electronics in the periphery, and it has been shown in [39] that the minimum frame readout time achievable with standard MAPS is in the order of 10 µs.

The ultimate goal should therefore be to replace the sequential processing of lines with an architecture where each pixel is able to detect that is has been hit and initiate its own sparsified readout.

Even with sparsified readout, the amounts of data to transfer per sensor will be on the order of gigabits per second, and fast serial transmission interfaces must be developed. With differential signaling and 8b10b encoding, where both clock, control and data are embedded into the same data stream, the data rate is increased while at the same time the number of wires to bond is reduced. Fewer wires facilitates bonding and gives lower material budget.

Means of development

Towards fulfilling the requirements of pixel detectors for CBM, different applications and experiments with less demanding requirements are targeted for the current MAPS based sensor designs. The requirements for three such applications are depicted in Table 1.1.

EUDET JRA-1 stands for a beam telescope developed as part of the EUDET project for detector research and development towards the international linear collider [45]. While the STAR experiment (Solenoid Tracker at RHIC) is an experiment built at the RHIC collider at Brookhaven with the purpose of studying the formation and characteristics of the Quark-Gluon-Plasma (QGP), a state of matter assumed to exist at sufficiently high energy densities [46].

The detector development is done step by step, starting with the most primitive and low bandwidth CMOS pixel sensors while making progress towards ever more complex and sophisticated detectors. The first step is to integrate data sparsification and serialized readout in a standard MAPS pixel matrix. Such a readout ASIC (MIMOSA26) was developed for the EUDET beam telescope. The second step is to construct an architecture that takes advantage of 3D technology to provide more intelligent pixels and to implement 8b10b encoding for fully taking advantage of serialization. Such a readout ASIC was developed in a technology exploration prototype for ILC.

The MIMOSA26 architecture was based on reading out the pixel matrix row by row and performing zero suppression by encoding groups of neighbouring hit pixels in each row. The parallell processing at row level, and the embedded zero suppression, has lead to both smaller frame readout time and smaller amount of data to read out per frame, ending at processing speed of 576 rows/100 µs. By increasing the clock frequency to 110 MHz, the frame readout time has been reduced further down to 83 µs.

However, this architecture is hardly scalable with the time resolution requirements of CBM. If the 576 row wide chip was replaced with two 256 row wide structures operating back to back, the readout time could be reduced to 42 µs. And by increasing the pixel pitch from 18.4 µm to 23.0 µm for the same fixed matrix size, one intends to reach 35 µs. Then finally by changing process technology from 0.35 µm to 0.18 µm and by using elongated pixels at one side of a detector station, one can come close to 10 µs frame readout time. But at some point, architectural

Table 1.1: Expected hit densities for the sensors of EUDET, STAR and inner layer of ILC. For STAR, two different hit densities have been modeled, each leading to two different sets of requirements.

Requirements	**EUDET JRA-1**	**STAR 200**	**STAR 500**	**ILC**
Hits per chip per frame	100	200	500	300
Pixel columns	1152	1024	1024	1024
Pixel rows	576	1088	1088	256
Hits per row (line)	2	3	4	6
ADC bits	1	1	1	1
Frame readout time	100 µs	200 µs	200 µs	25 µs

limitations make it too difficult to further reduce the frame readout time, and 3D technologies must be utilized for taking the frame readout time to even smaller values.

At IPHC, three different 3D architectures have so far been developed:

The first architecture resembles that of a strip detector, and therefore with a usability limited to applications with low multiplicity - like a beam telescope. In the second architecture, targeted for ILC, individual particles trigger the latching of timestamp information in the individual pixels, and the timestamp information is read out of the chip in the 199 ms dead time between the 1 ms bunch trains of ILC. In the third architecture, a rolling shutter operated pixel matrix has been 3D integrated with the purpose of providing a binary discriminator inside every pixel cell - instead of the shared column-level discrimators used in previous rolling shutter chips. Removing the shared discriminator improves the performance and may reduce the row processing time from 160 ns in MIMOSA26 down to 40-80 ns. However, it requires a new kind of readout architecture to interface the pixelized matrix of discriminator outputs.

For a high luminosity experiment like CBM, where continuous readout is required as there is no dead time between bunch trains, neither the first nor the second architecture are suitable. However, the third architecture, which results from a cooperation between IRFU in Saclay and IPHC in Strasbourg, has become the starting point for a conceptual design targeted at CBM.

Chapter 2

Read out architectures for particle trajectory detectors

In the previous chapter, different types of pixel detectors were presented. And it was concluded that MAPS currently is the only kind of technology that is able to meet the CBM and ILC requirements of both high spatial resolution and low material budget - although technological and architectural improvements are necessary to meet the requirements on time resolution and radiation hardness.

The frame readout time is the time required to electronically read out all the samples of a pixel matrix. The readout rate requirements are ideally governed by the physical interaction rate, usually the rate of hard collisions in the experiment. However, in many detectors, like with the ILC, the intense beamstrahlung makes it necessary to read out the accumulated charges at a much higher rate than the actual interaction rate - otherwise, the interesting physics is drowned in all the background. The same case applies for the CBM experiment, where heavy ions collide with a target representing $\approx 1\%$ interaction length, and where the largest hotspots are due to delta electrons and not due to actual heavy ion collisions.

With CBM, a frame readout time of typically $2\,\mu\text{s}$ is desired for keeping the occupancy at an acceptable level. Due to the reduced frame readout times required for ILC and CBM, and the correspondingly increased raw data rates, it is necessary to develop faster, sparsified readout architectures for MAPS.

In this chapter, we start by presenting clusterization as a way of modeling hits due to particles, and how these clusters appear in a MAPS based pixel detector with binary discrimination. Then different kinds of sparsified readout architectures are presented and their compliance with MAPS is discussed.

2.1 Physical model - clusters

The purpose of using pixel detectors in projects like ATLAS, ILC, and CBM, is to measure the trajectory of minimum ionising particles coming out from the interaction point of the experiment. If the pixel detectors are built up layer by layer around the interaction point, the tracks of the particles can be reconstructed by making a fit of the different impacts in the pixel matrix where the particles have passed. As the detector is supplied with a magnet to provide magnetic deflection of the particles, it is even possible to obtain the momentum of the particles using this approach.

As long as the charge collection electrodes are able to collect a number of electrons that is significantly above the equivalent noise charge (ENC) of a pixel, a hit may be detected. With MAPS sensors, the most probable value of the SNR for a minimum ionising particle ranges from 15 to 30, depending on the pixel size. Detection efficiencies exceeding 99.5 percent have been demonstrated even in the case of a pitch as large as 40 μm [36].

With the simplest pixel architecture, a single bit is used to indicate if a pixel cell has been hit. We may therefore model the pixel detector as a binary matrix where each element in the matrix has a zero value if no particle has passed through it - and a binary one value if a particle has passed through it.

In Figure 2.1, we see a particle passing diagonally through the substrate of a planar pixel detector. For a planar and fully depleted silicon device, with an active volume that typically could be 300 μm thick, we could assume that the energy deposited by the particle was roughly identical to the mean energy given by the Bethe-Bloch formula, and that the number of generated electrons would correspond roughly to one electron per 3.6 eV of deposited energy. We could also assume that the induced charges would drift vertically in the electric field and become captured by the pads located above.

The number of pixels affected by an impinging particle depends on the pitch of the pixel cell. If the pads are small, one might assume that the charge of a particle will spread on a number of neighbouring pixels - how many is dependent on the pixel pitch. But even if the pads are relatively large, there will always be cases where the charge of a particle is spread onto two neigbouring pixels.

We can therefore consider the binary matrix to be an ocean of zeroes, but with with islands composed of neighbouring ones at the locations where there has been a hit by a particle. These islands will be of varying size and they are called clusters. In addition to the clusters, there will also be randomly distributed ones due to noisy pixels.

With pixel sensors based on MAPS, the active volume consists of a maximally

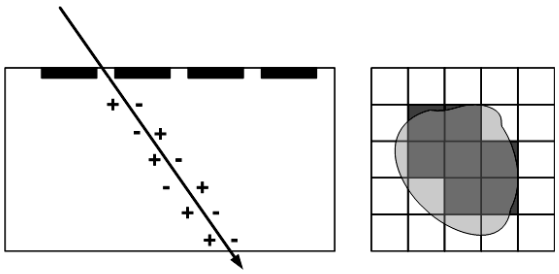


Figure 2.1: Particle passing through substrate of a planar pixel sensor. Side view to the left, showing the particle passing below different pads, and top view to the right, showing the clusterised manifestation in the pixel matrix. We also notice that one of the pixels in the cluster is neigbour to all the other pixels in the clusters - the central pixel.

20 and typically 14 μm thick epi layer, which is much smaller than the hundreds of micrometers available in a fully depleted device. With such a thin layer, there is not only a much lower amount of average energy deposited in the active volume, but also the Landau variations in deposited energy become a much larger fraction of the overall deposited energy. And with the active volume being undepleted, there is no drift mechanism for passing the generated electrons towards their collecting electrodes. Instead they are diffusing randomly underneath the pixel cells before finally being collected on a diode. This transport mechanism does not only result in reduced charge collection efficiency due to recombination, it also gives a spatial smearing of the signal. For the same minimum ionizing particle passing through a MAPS sensor, we therefore do not know if the result is a large cluster or simply a single hit pixel. This point becomes important later in this chapter, when discussing data reduction techniques. Since all kinds of patterns must be taken into consideration, only lossless data compression algorithms are suitable for MAPS sensors.

The single point resolution of a purely binary matrix without any clusters is the pixel pitch divided by the square root of twelve ($\frac{p}{\sqrt{12}}$) [3]. However, the single point resolution may be improved even more if a hit results in a cluster - whose gravity center may be computed with a weighted sum of the charges in the hit pixels [3]. The larger the number of bits that are used for quantifying the deposited charge in a pixel, the larger becomes the spatial resolution - until finally being limited by the SNR of the pixel.

For pixel detectors based on MAPS, simulations and experimental data show that applications with 20 μm pixel pitch and epitaxial depth of 12-16 μm give clusters that are localized within 3x3 pads [1]. And test results for the 18.4 μm-pitch

MIMOSA26 show that, thanks to the cluster shape of the signal, single point resolutions below 4 µm can be obtained for MAPS-based sensors with similar pixel widths, even when the pixels do not have any other quantization of charge than simple binary discrimination (see for example [47]).

Calculation of the gravitational centers of the hit pixels, called clusterisation, is a technique that has already been utilised in previous high energy physics experiments. When the different clusters belonging to a particle are identified, the vertex of the particles is reconstructed by track fitting of its clusters. The computation of clusters has traditionally been performed offline, but it is in principle also possible to perform online. Online clusterisation is useful for two reasons:

- The data stream from the detector is reduced by suppressing all unaffected pixels outside of a cluster (lossless encoding).
- Interesting data (clusters of hits due to particles) may be identified and encoded as tracks of particles while uninteresting data (randomly spread noisy pixels) may be rejected and suppressed (lossy compression).

An example of online clusterisation could be the High Level Trigger (HLT) for the ALICE experiment at LHC [48]. For the ALICE Time Projection Chamber, data from the individual sensors (frontend cards) are collected by a readout control units and transferred fiberoptically to readout receiver cards with a PCI interface. The computers containing the readout receiver cards are part of a distributed computer network called the high level trigger, which is to perform online track reconstruction before the reconstructed tracks are sent to a storage medium. The first stage of the track reconstruction is the cluster finding, which in the case of HLT is performed on a dedicated FPGA inside the Readout Receiver cards - before the clusterised data is DMA-transferred to the computer memories via the PCI interface.

For an experiment like CBM, which is also to be equipped with a High Level Trigger [49], it may be necessary to implement online clusterisation inside the individual sensors, with the purpose of reducing, to an acceptable level, the amount of data handled by the data acquisition system. To be able to perform such clusterisation, the electronics of the sensor must be capable of localizing and time tagging the individual clusters before they are overlapped by clusters coming from new hits.

2.2 Readout and data sparsification architectures

The readout and data sparsification architectures for pixel detectors can be distinguished on whether they provide centralized readout or data-driven readout. With centralized readout, all the pixels in the matrix are being read out from the periphery of the pixel matrix and then eventually processed by a data sparsification algorithm before being transmitted to the data acquisition system. While with data driven readout, only the pixels containing hits are being read out, with the latter requiring that the logic for initiating readout is distributed to the individual pixels.

2.2.1 Centralized readout with rolling shutter

The simplest way of reading a pixel matrix is to read the pixels one by one, from the left to the right in every line. The first MAPS based MIMOSA chips were implemented in this way [1].

To improve the readout speed, and hence the time resolution, the readout may be parallellized by reading all the pixels in a row at the same time. This has been the approach with the latest series of MAPS based MIMOSA chips, like MIMOSA22 and MIMOSA26, with the latter to be further described in Chapter 3.

The two approaches are illustrated in Figure 2.2.

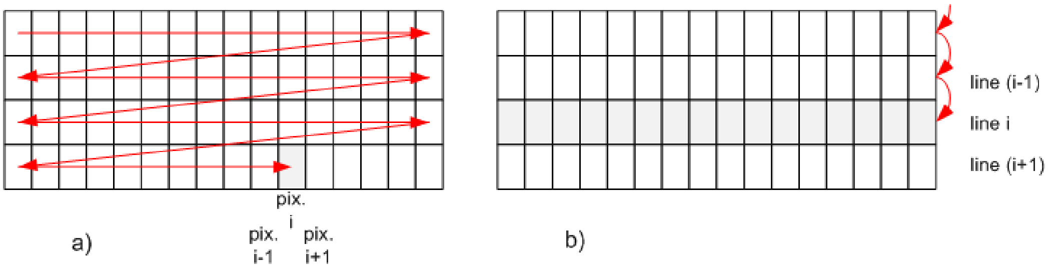


Figure 2.2: a) Readout pixel by pixel (early MIMOSA chips). b) Readout line by line (MIMOSA26)

Principle of operation

With rolling shutter operation, each row, called line, is processed at a fixed time interval. The lines are read from top to bottom, and then the process is repeated. A sequencer activates the different lines one by one. The time to read all the lines in a matrix gives the time resolution of the sensor.

[1]With the first MAPS based MIMOSA chips, each individual pixel was externally addressed and read out using two shift registers, one for row addressing and another one for column addressing

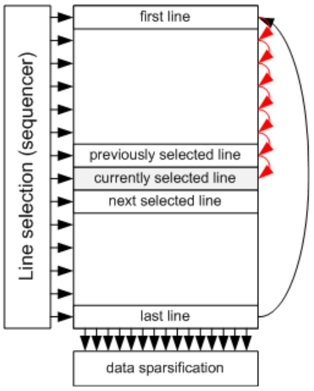


Figure 2.3: Block schematic of rolling shutter operated pixel matrix with data sparsification circuitry in the periphery. The preamplifiers of each line is powered on and the pixel signal read out at each their time slot. While the discriminators and the readout electronics in the periphery are shared for all the lines and always being active.

Data sparsification algorithms

As the readout electronics of a rolling shutter operated sensor only sees the pixel matrix line by line, a data sparsification algorithm also has to analyze the pixel matrix line by line, or eventually implement a pipeline for analyzing the patterns of hits in both dimensions.

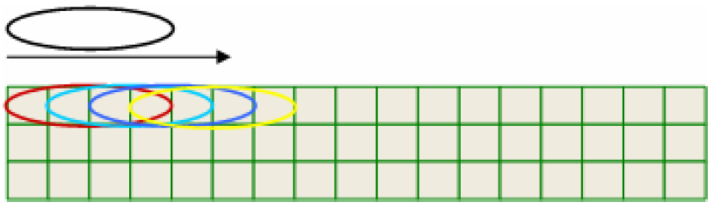


Figure 2.4: Combinatorial search for states within the state encoder. A sliding window of four pixels identifies groups of hits, called states.

For a one-dimensional line analysis, a combination of sparse scan and combinatorial state encoding has been utilized in MIMOSA26 to identify and encode patterns of hits in a line. The sparse data scan (SDS) operates as a sliding window, processing the individual line from left to right and presenting each pattern of hits by providing the address of the leftmost pixel in a group and an encoding for the number of succeeding hit pixels. Its asynchronous principle of operation is shown in Figure 2.4.

As the time for processing each line is fixed, the maximum number of hits to extract from each line must also be fixed. And even if the average number of hits in the matrix is low, the Poisson shaped distribution of hits may cause overflow situations where there are more hits in a line than the readout electronics is dimensioned for handling. The data sparsification circuitry must therefore be dimensioned with respect to the highest tolerated probability of missing a hit in a line.

Figure 2.3 shows a block diagram of a rolling shutter operated MAPS sensor, with readout and data sparsification in the periphery.

2.2.2 Data driven and triggered readout with timestamping

An alternative to centralised processing is data driven distributed processing, where each pixel cell independently detects that it has been hit and initiates readout. With this approach, readout and sparsification are combined.

To improve the time resolution, each pixel cell may be supplied with a counter that is running freely until a hit is detected. If the counter is synchronous to the bunch crossing clock[2] of the experiment, it is possible to assign every single cluster to a specific bunch crossing.

An efficient and logically simple way to implement data-driven readout is to implement a sparse scan token ring between the pixels in each column. The first pixel that grabs the token is allowed to send data, passing the token on to another pixel on the next clock cycle. The FE-I3 readout chip for the ATLAS pixel detectors is an example of a design implementing such a structure [43]. The pixels in each column share a token ring and a data bus, where the pixel holding the token places its hit information on the data bus. The information placed on the data bus is in the next turn stored in hit buffers at the chip periphery, and the readout of the buffers is finally initiated by the ATLAS Level 1 Trigger. The hits with timestamps corresponding to the triggered event are then read out and the rest are rejected.

The main problem with the token passing scheme is that the readout speed is limited by the time the token needs to ripple down the entire row of channels. One way to improve the timing is to implement some kind of fast look-ahead logic in addition to the ripple logic. An example of such an implementation is the priority look-ahead logic implemented for the ATLAS pixel front-end electronics. A fast priority scan path is implemented by minimizing the delay in the critical data path. The 160 pixels in a column are arranged in 10 banks. Each bank consists of 16 pixels that come in three different types. The first two types of pixels consist of NAND

[2]In collider experiments, the particles that are to collide are organized into larger bunches of particles - which are synchronised to arrive at the collision point at the same time.

and NOR cells for rippling the priority during the sparse-scan. The pixels alternate between the NAND and the NOR version to minimize the priority sparse-scan delay, enhancing the readout speed. And at the end of each bank there is a third type of pixel cell connected to a look-ahead path for a fast priority ripple [50].

Another example is the BUSY logic utilised by the FE-I3 readout chip for the (both planar and 3D) ATLAS pixel sensors. To increase the speed, a column is divided into ten 16-cell blocks, where one fast BUSY signal is calculated on the block level, and the block-level BUSY OUT is the OR function of the block-level BUSY IN and the cell-level BUSY OUT [43].

Compared to sequential line processing with a fixed frame readout time, the data driven approach is especially efficient with respect to readout time if the percentage of occupied pixels during a frame is low. Overlapping between hits is avoided by immediately transferring hit information out of the matrix. To avoid the loss of second hits while waiting for priority to read out, the individual hit pixels can also contain buffers for eventual second hits - or at least an overflow bit to indicate that such a hit has occurred.

Compared to continuous readout, triggered readout is an effective mean to reduce the required bandwidth of the sensor readout circuitry - provided that the buffers in the chip periphery are sufficiently deep to store the hit information of triggered events arriving close to each others in time. Otherwise the buffers will in some cases overflow and a dead time is in practice introduced.

2.2.3 Data driven and delayed readout with timestamping

A major weakness of sequential row processing is that it is impossible to distinguish between overlapping clusters arriving within the same frame. However, if the individual pixel cells had been provided with time tagging capabilities, they could be distinguished by their time stamps.

While two overlapping clusters are indistinguishable with sequential processing, they are easily distinguished by their time stamps. If the pixels cell are equipped with an overflow bit to register secondary hits between the initiated readouts, the time stamping of hits even makes it possible to fully reconstruct partly overlapping clusters (Example: Cluster no. 1 with timestamp A has cluster no. 2 with timestamp B as its nearest neighbour. The pixels of cluster no. 1 that are close to cluster no. 2 are also tagged for a second hit. We therefore know that these hits most likely belong to cluster no. 2.).

A way to solve the problem could therefore be to make the resolution of the pixel matrix high enough to make sure that the individual clusters only partly overlap. And instead of continuous readout, the hits could simply accumulate until the entire bunch train had passed. Afterwards, the readout electronics would have the entire dead time between the bunch trains to read out the content of the matrix. Examples of experiments where such an approach is possible could be the ILC and CLIC, where particle bunches are grouped in trains that are separated by a relatively long, beam free, time interval.

An example of a sensor that simply accumulates the number of hits until the content of each column is read out as one long shift register is the single-photon counting Medipix chip [51] (55 µm pixel pitch). Although the Medipix chip simply counts the number of hits within a specified energy window without providing any kind of timestamping, the concept could easily be expanded by providing timestamping registers in each pixel cell.

In the SDR0 prototype chip for ILC, implemented in DNW-MAPS with pixel pitch of 25 µm, each pixel contains a discriminator and a 5-bit timestamp register, with the timestamp values provided by a grey counter located in the chip periphery. Each pixel also contains token passing circuitry for readout. During the bunch train, the discriminator triggers the latching of timestamp values provided by the grey counter. While between the bunch trains, the chip is operated in a delayed readout mode where a token is passed through all the pixels, and where the pixels containing hits grab the token and pass their timestamps to the chip periphery [52].

2.2.4 Architectures suitable for MAPS

Of the above mentioned architecture, centralised readout with sequential row processing is so far the only architecture implemented in standard MAPS. While data driven architectures with time stamping are the architectures of choice for hybrid pixel detectors, MAPS does not have sufficient area per pixel cell, nor the availability of PMOS transistors, to implement such a scheme.

If the entire matrix is to be scanned row by row, there will be a fixed integration time for the entire image. And the occupancy[3] of the pixel matrix will be a function of particle flux, pixel multiplicity, and integration time. Reducing the integration time is therefore a mean to reduce the occupancy of the sensor. For the ILC, the maximum integration time is set to 25 µs due to the beam background. For the CBM, the desired time resolution is as low as a few microseconds, due to delta electrons being knocked out of the target. As will be shown in Chapter 5, new technologies based on 3D integration have to be explored for meeting the requirements for such a frame readout rate.

[3]The occupancy of the pixel matrix is the percentage of fired pixels in the matrix.

2.3 Data reduction techniques

To successfully perform the particle trajectory reconstruction, most experiments require that the occupancy of the pixel matrix is kept below a few percent. The higher the flux of particles becomes, the shorter frame readout time is therefore required.

Reducing the frame readout rate increases the raw data rate. But since the high frame readout rate is performed to ensure a small fraction of hit pixels, the amount of data to transfer could be greatly reduced in size by utilising some kind of loss-free data compression where the unaffected parts of the matrix are suppressed and only those parts of the matrix that contains hits are addressed and encoded. In fact, if each hit pixel or group of hit pixels was uniquely encoded with its full row and column address, the data rate of the sensors would only be dependent on the particle flux and no more on the frame readout rate (in the idealized case when noisy pixels and overlapping hits are ignored).

Two possible methods of hit encoding have been proposed for binary discriminated MAPS:

- One-dimensional state encoding, processing the image row by row while addressing and encoding small line segments of binary ones.
- Two-dimensional clusterization, processing the image by sliding over three rows at a time and addressing and encoding clusters of hits.

2.3.1 State encoding

With state encoding, the discriminated pixel matrix is processed line by line with the segments containing hit data being encoded and addressed. This is done by creating small data packets, where each packet contains the column address of the first hit pixel and the state encoding of the associated line segment. The concept is further illustrated in Figure 2.5.

Such a circuit for identifying and encoding states has been developed in AMSC35 technology for MIMOSA26 (by X. Fang and A. Himmi). A block schematic of the circuit is shown in Figure 2.6, with a corresponding timing waveform shown in Figure 2.7, and the tasks of this circuit are to be described in more detail.

The state encoding circuitry is part of a pipelined datapath. For each new line to be analyzed, the CKLATCH clock registers the pixel inputs of current line and the encoded state outputs of previous line. The fixed line processing time is thus equal to the period of CKLATCH (160 ns in MIMOSA26).

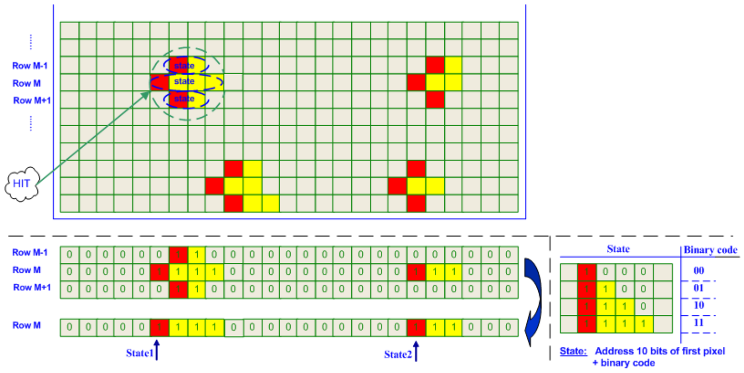


Figure 2.5: State encoding of a regular cluster.

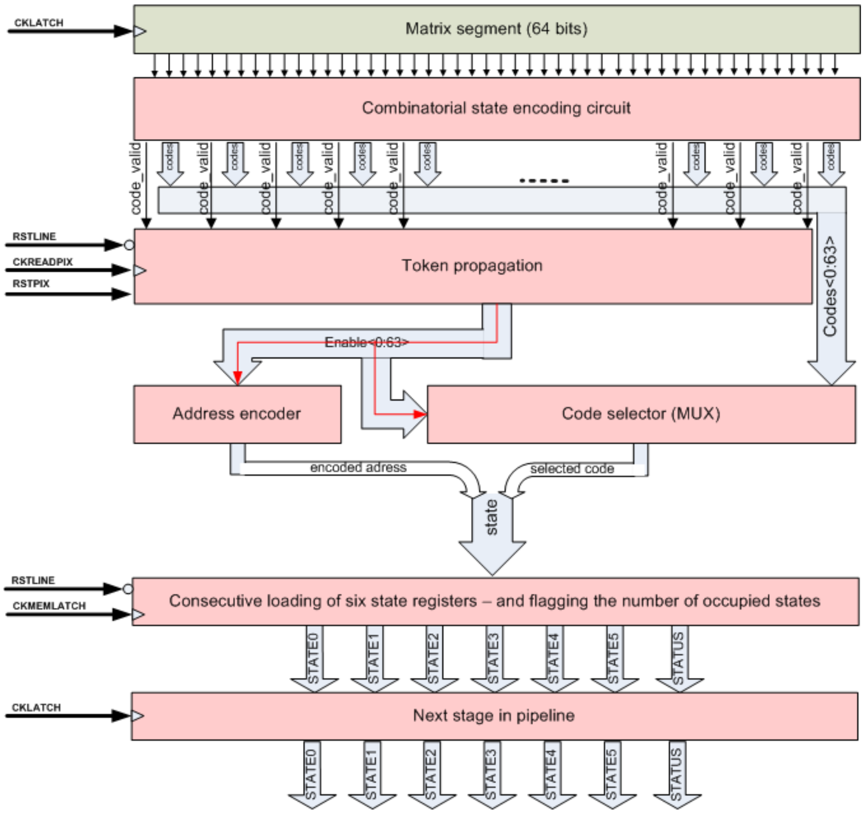


Figure 2.6: Block schematic of the state encoding circuitry [53].

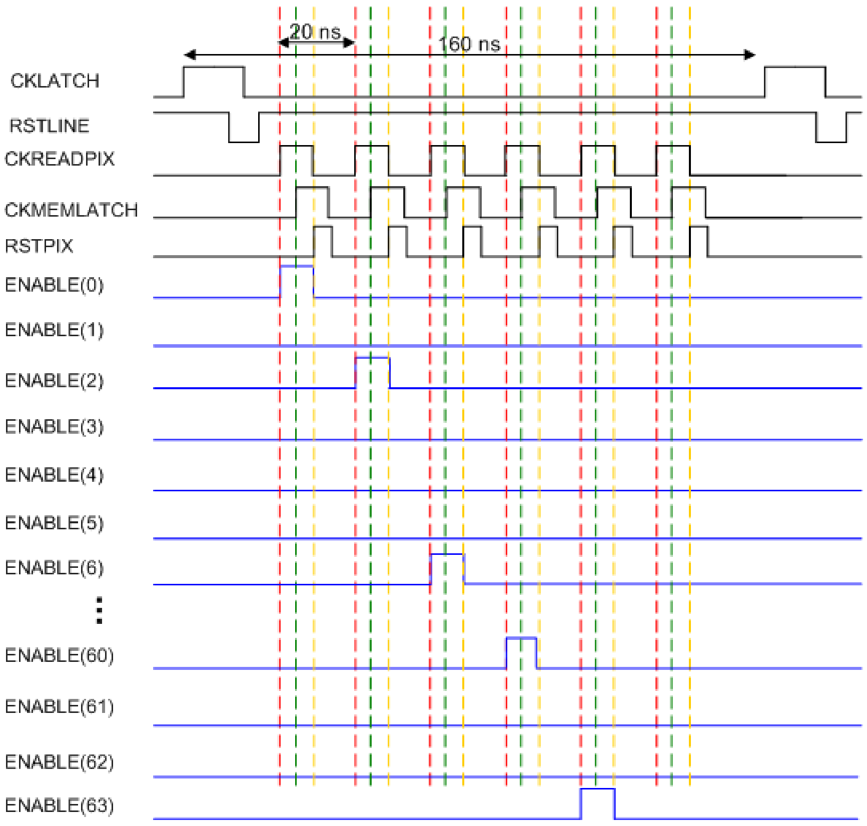


Figure 2.7: Timing waveform for the state encoding circuitry.

After being latched into the input register, the binary discriminated pixels first enter a combinatorial state encoder, going through the individual bits of a line to identify every pixel that corresponds to the first bit in an encoded state. For each pixel identified as the first pixel of a state, a validation signal is asserted, together with the corresponding state encoding for every such pixel.

To one by one select the valid states, a sparse scan chain is used for sweeping a token asynchronously through all the validation bits from left to right until being blocked by the first pixel pointing to a valid state. The encoded state corresponding to the identified validation bit is then selected. To optimize the speed of the token, the scan path is implemented as a chain of intertwined NAND and NOR gates, whose implementation details are to be further described in Chapter 3.

As seen from the waveform of Figure 2.7, the sparse scan chain is synchronous to a pixel clock, called CKREADPIX, and for each rising edge of CKREADPIX, the pixel cell blocking the token gives out an enable bit. The enable bit is in the next turn used as input for encoding the corresponding pixel address and selecting the corresponding set of encoded states. The state address and the code is then stored into a register which is sensitive to another clock, called CKMEMLATCH (created by delaying CKREADPIX with 5 ns), and then a RSTPIX signal is asserted for propagating the token to the next pixel with valid state information.

On the next rising edge of CKREADPIX, the sparse scan chain proceeds with identifying the next pixel that has stopped the token. There are six CKREADPIX clock pulse during the processing one line, and up to six states can therefore be identified. A status flag is asserted to indicate the number of state outputs that contain valid states.

2.3.2 Clusterization

The clusterization algorithm analyzes the image in both horizontal and vertical direction, and for each identified cluster, it gives out the address of its central pixel, together with an encoding of the pattern of surrounding pixels.

Assuming that standard MAPS has a regular cluster size of dimensions 3x3, it is sufficient to monitor three rows at the same time. A window of three rows, sliding over the image, is implemented by feeding the discriminator outputs into a three-stage pipeline and look for central pixels located in the second stage of the pipeline.

In MAPS, it is assumed that a regular cluster is composed of a central pixel and four crown pixels, as shown in Figure 2.8. To find out if a pixel is a central pixel in a cluster, it is therefore necessary to examine the individual pixels together with their two nearest neigbours - both vertically and horizontally. In theory, diagonal examination could also be needed to identify rare clusters, but experimental data shows that the contribution from such clusters are small and that it has marginal consequences to ignore the pixels in the corners of any complete or partial 3x3 cluster. A simple function for finding central pixels of regular clusters could therefore be:

$$en_{ij} = en\_c_i \cdot en\_l_j,$$

where en_c_i is a function that analyses pixels of the actual line, and en_l_j is a similar function for pixels of the actual column.

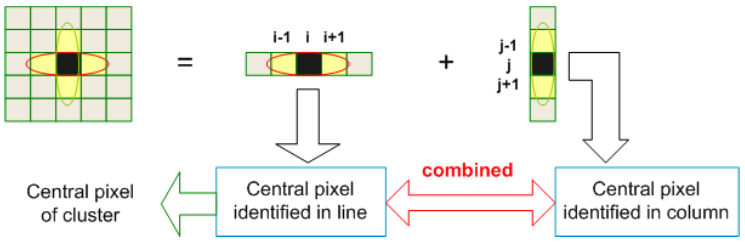

=
i-1 i i+1
+
j-1
j
j+1
Central pixel of cluster
Central pixel identified in line
combined
Central pixel identified in column

Figure 2.8: Decomposition of a cluster [53].

As seen from the formula, a cluster is identified by analyzing lines and columns separately, and combining the results.

The function for identifying a central pixel inside a line can be decomposed into

$$en\_c_i = Pcol_i \cdot Pcol_{i-1} \cdot Pcol_{i+1},$$

where $Pcol_i$ is the binary discriminated value of the pixel in column number **i** inside the analysed line.

Similarly, the function for identifying a central pixel inside a column can be decomposed into

$$en\_l_j = Pline_j \cdot Pline_{i-j} \cdot Pline_{j+1}.$$

where $Pline_i$ is the binary discriminated value of the pixel in line number **i** inside the analyzed column.

As shown in [53], the final functional equation when irregular clusters and overlap between clusters are taken into consideration, finally proves to be:

$$en_{ij} = en\_c_i \cdot en\_l_j + en\_l_j \cdot \overline{(en\_c_{i-1} \cdot en\_l_{j-1})} \cdot \overline{(en\_c_{i+1} \cdot en\_l_{j+1})}$$

where $en\_c_i = Pcol_i \cdot (Pcol_{i-1} \cdot \overline{(en\_c_{i-2})} + Pcol_{i+1}) \cdot \overline{(en\_c_{i-1})}$

and $en\_l_j = Pline_j \cdot (Pline_{j-1} \cdot \overline{(en\_l_{j-2})} + Pline_{j+1}) \cdot \overline{(en\_l_{j-1})}$.

A block schematic of a clusterization circuit, designed and implemented by X. Fang, is shown in Figure 2.9 [53]. The outputs of the combinatorial cluster finder are input to the same kind of sparse scan algorithm as used in the state encoding case, with the sparse scan presenting the different central pixels one by one to the address encoder and cluster code multiplexer. Using the same clocking scheme as for the state encoding, up to six encoded clusters are extracted and fed to the next stage in the pipeline.

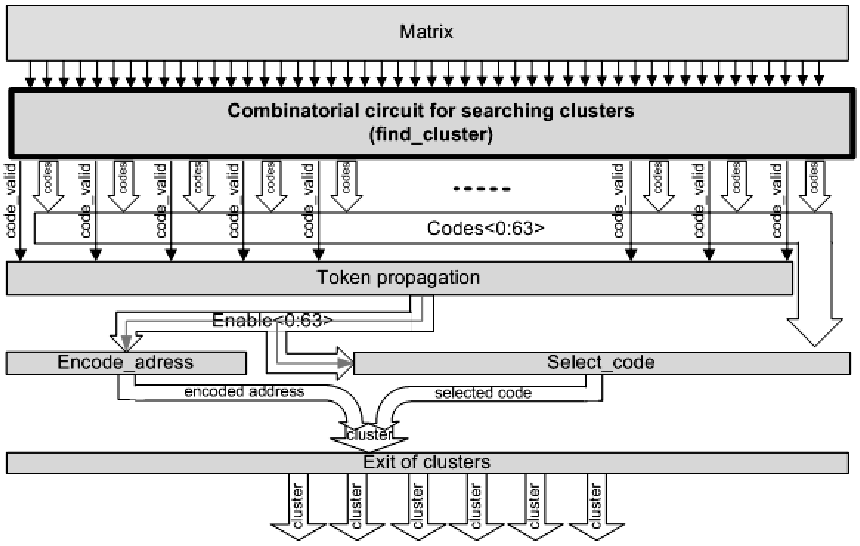


Figure 2.9: Block schematic of the cluster finding and encoding circuitry [53].

It should be noted that the clusterisation circuit presented above is completely loss-free. Even a single hit pixel with no neighbouring hit pixels is modeled as a cluster.

2.3.3 Compression efficiencies in standard MAPS with moderate occupancies

If we assume a matrix of dimensions 1024x1024, 10 bits are required for column addressing. Ignoring the row addressing, each cluster is therefore of 18 bits, and each state of 12 bits. In addition to the hits due to particles, there are fake hits due to the noisy pixels.

If we assume standard MAPS with a hit multiplicity of 5 (regular cluster), and a noisy pixel fraction of 10^{-4} (worst case assumption, based on beam tests of MIMOSA26 [54], although pixel designs made in the same AMSC35B4 process can be tuned to obtain fake hit rates below 10^{-5} [38]) then the compression efficiency of state encoded zero suppression (3 states/hit and 1 state per noisy pixel) can be written as:

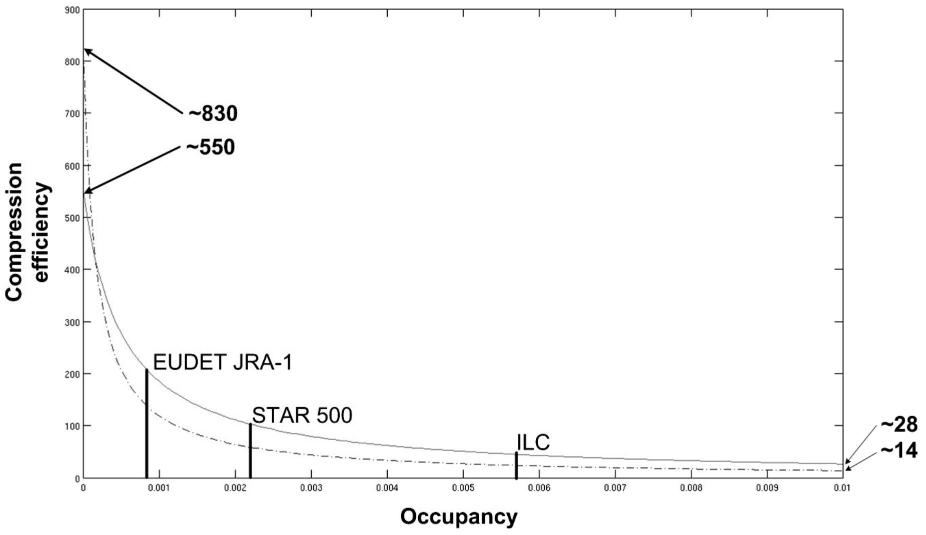


Figure 2.10: Compression efficiencies of state encoding (dash-dotted line) and clusterization (solid line) shown as functions of occupancy (ranging from zero to one percent). The occupancies of EUDET-JRA1, STAR 500 and ILC are inserted with separate markers.

$$\frac{pixels/chip}{hits/chip \cdot 3\ states/hit \cdot 12\ bits/state + pixels/chip \cdot 10^{-4}\ fakes/pixel \cdot 1\ state/fake \cdot 12\ bits/state}$$

Substituting hits/chip with ($occupancy \cdot pixels/chip/5$), and further manipulating the expression, the compression efficiency is finally expressed as a simple function of occupancy:

$$Compression\ efficiency = \frac{1000}{occupancy \cdot 7200 + 1.2}$$

For clusterization, one can make the similar calculations with one encoded cluster for each particle hit and one for each noisy pixel, and arrive at the following expression:

$$Compression\ efficiency = \frac{1000}{occupancy \cdot 3600 + 1.8}$$

Figure 2.10 shows the compression efficiencies plotted as functions of occupancy, with markers pointing out the occupancies of EUDET JRA-1 (0.08%), STAR 500 (0.22%) and ILC (0.57%). One sees that for the occupancies of these experiments, the compression efficiency is almost twice as good for clusterization as for state encoding. But when the occupancy goes as low as to 0.002 percents, the fake hits starts to dominate, and the compression efficiency satures at around 550 for clusterization and around 830 for state encoding.

For a given particle flux, fake hit rate, and frame readout time, the data rate of a zero-suppressed sensor with state encoding can be written as:

$$Data\ rate = \frac{(frame\ readout\ time) \cdot hits/chip/sec \cdot 3 \cdot 12 + noisy\ pixels/frame \cdot 12}{frame\ readout\ time}$$

Keeping the experiment particle flux fixed, Figure 2.11 shows the data rates plotted as functions of frame readout time for the EUDET JRA-1, STAR 500 and ILC pixel designs. The figure confirms the observations of Figure 2.10. Within the interval of moderate occupancies, a reduced frame readout time is compensated with an improved compression efficiency, and the data rate is kept more or less constant. But at some level the fake hits start to dominate and the data rate becomes once again proportional with the frame readout rate.

However, test results of MIMOSA26 at frame readout times of 460 µs
($\approx$ 90 $noisy\ pixels\ per\ frame$), 115 µs ($\approx$ 40 $noisy\ pixelsperframe$) and 90 µs ($\approx$ 30 $noisy\ pixels\ per\ frame$) show that the fraction of noisy pixels is reduced with the frame readout time [55]. The reasons of this dependency have not been clarified, although it is a reasonable to assume that a higher frame readout rate reduces the low frequency contribution to the rms noise (like for example the 1/f noise). The expected amount of noisy pixels at even lower frame readout rates is therefore not known (personal correspondance with M. Goffe).

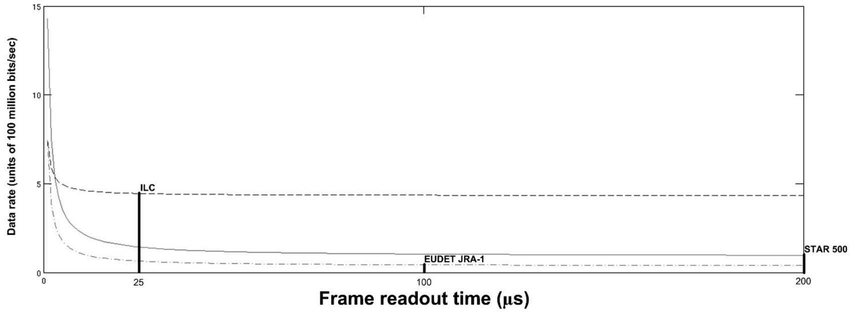


Figure 2.11: Data rates with state encoding shown as functions of frame readout time for the fixed particle fluxes of EUDET-JRA1, STAR 500 and ILC.

Although the presented near-future designs are targeted for occupancies where the contribution of fake hits to the data rate is small or negligible compared to the contribution of particles, a strategy for suppressing fake hits can still become important to reduce data rate in future designs with much smaller frame readout times, and hence occupancies, than those of current pixel designs.

2.3.4 Compression efficiencies with respect to multiplicity and matrix width

The efficiency of clusterization with respect to state encoding depends on the number of bits in the column addresses and the multiplicity of the hits due to particles.

If the ratios between the number of words occupied by three states and those of one encoded cluster are plotted as functions of the column address length, the asymptotic values would be $\frac{6}{8}$ for no addresses at all and 3 for an infinite address length. For a reasonable number of column address bits like 10, the ratio is 2.

As long as one assumes a regular cluster of size 3x3 pixels, one will have three encoded states for each encoded cluster, and clusterisation will therefore give a significant data reduction compared to state encoding. However, with detector technology that gives clusters of smaller size, like 2x2, there will only be one or two encoded states for each encoded cluster.

If 10 bits are used for column addressing (1024 columns in the pixel matrix), which give encoded states of 12 bits and encoded clusters of 18 bits, an average of 1.5 states per cluster will give exactly the same amount of data for both encodings.

While the sparse scan and state encoding circuitry has been implemented in layout with a total height of 310 µm, the sparse scan and cluster finder circuitry has been implemented in layout with a total height of 500 µm. When implemented in hardware, the clusterization algorithm for binary discrimination therefore results in a slightly reduced active surface of the detector.

2.3.5 Comparison of clusterization and state encoding with general compression techniques

Until the actual particle trajectories have been reconstructed by the high level trigger of the data acquisition system, it is necessary to solely perform lossfree data compression, as a single particle can result in anything from a single hit pixel a large cluster.

According to general information theory, the loss-free compressibility of a data stream is dependent on the entropy of its source. To evaluate if a compression algorithm is optimal with respect to a given source of data, one should therefore calculate the entropy of the source, and then compare the code rate of the compressed code with the theoretical minimum value given by Shannon's source coding theorem [56].

For a quantitative analysis of the compression efficiency of clusterization and state encoding in MAPS based pixel sensors, it would thus be necessary to start with analyzing data from actual experiments, calculate their entropy, and compare the code rates of the compressed data with the theoretical minimum value.

However, no such analysis has been performed on the data of MAPS based pixel sensors. To give a judgement on the compression efficiency of clusterization, a qualitative comparison was performed between clusterization and JPEG, with the latter being a standard technique for compressing images.

An image compressed with JPEG is taken through the tree steps of Discrete Cosine Transform (DCT), run length encoding (RLE) and Huffman encoding. The entire comparison between clusterization, state encoding and JPEG is shown in Appendix B, and the main conclusions are that:

- One cannot take advantage of the initial lossy step of DCT because all shapes of clusters are expected and no lossy compression techniques are allowed.
- The next step of Run Length Encoding is practically speaking the same as state encoding, with the latter being an improved implementation with the expected shape of the pixel matrix taken into consideration. Clusterization is even more efficient as it encodes into one addressed cluster what would otherwise be three addressed states.

- The last step of building a Huffman tree is efficient if some characters are more frequent than others. As the localization of hits must be considered is completely random, the addresses of the clusters are also completely random, and the compression efficiency of building the Huffman tree is therefore small. Building such a tree optimally also requires that the image is analyzed in its entirety, instead of line by line, and such a feature would therefore require more logic and larger memory.

It therefore turns out that the interesting RLE features of JPEG already are available with state encoding and clusterization, with the other features either not desirable (DCT) or requiring more logic and memory while only giving minor improvements in efficiency (Huffman encoding).

The best argument for utilising clusterisation is, however, that it is based directly on a physical model, and the clusters extracted in the sensor readout electronics can be used directly by a track fitter to reconstruct the particle trajectories. This would not be possible if conventional, lossfree techniques were utilised. In such a case, the image would first have to be decompressed, and then cluster finding, Hough transform or some other kind of vertex reconstruction technique had to be run afterwards. When clusterisation is used as method of data reduction, we have the benefit of performing this first stage of data processing already at a sensor level, thus reducing the computing load on the data acquisition system (if state encoding is used, the data acquisition software has to combine the states into clusters before performing the track fitting).

2.4 Segmentation of pixel matrix into sparse scan banks

2.4.1 Necessity of segmentation

With the state encoding circuitry previously described, two critical paths are identified:

- The path of the token going from the left to the right through the sparse scan chain. For a line width of N pixels, the token must in the worst case propagate through N/2 gates of NAND and N/2 gates of NOR.
- The path through the multiplexer that selects the states of the combinatorial state encoder. For a line width of N pixels, an N-to-1 multiplexer is implied.

If the width of a line is doubled, it means that the scan path becomes twice as long, and the multiplexer becomes twice as large.

The delay through the scan path gives the minimally allowed period of the token passing clock, while the hit density of the experiment gives the maximum number of states to extract from a line, and thus the minimally required period of the token passing clock. At some point, either with increasing hit density or with increasing line width, the required clock frequency becomes higher than the maximally allowed clock frequency, and it becomes necessary to split the circuitry into separate banks.

The outputs of the different segments must in the next stage be collected by a multiplexer structure that is dimensioned for selecting as many states as have to be taken into account for the entire line.

2.4.2 Calculating the width of the segments

With the state encoding circuitry previously described, up to six states are extracted during the processing of one line. As long as a regular cluster of 3 x 3 dimensions is assumed, there will be three encoded states for each hit, and the state encoder is therefore capable of processing matrices with hit densities up to 2 hits per line without missing hits.

Knowing the maximally allowed number of hits per segment, the maximally tolerated segment width for a given hit density can be calculated.

The probability of having exactly N hits in a line segment is given by the Poisson equation,

$$P(N,\lambda) = \frac{\lambda^N \cdot e^{-\lambda}}{N!},$$

where λ is the average number of hits per line.

The average number of hits per line can be calculated as:

$$\lambda = \frac{(hits/s/chip) \cdot (frame\ readout\ time)}{(sensor\ height/pixel\ pitch)}.$$

To set the maximum number of detectable hits per line, we need to know the probability for a number of hits that is larger than $N = N_{MAX}$. This probability can be calculated by discrete summation of all the probabilities from N=0 to $N = N_{MAX}$:

$$P(N > N_{MAX}) = P - P(N \lesssim N_{MAX}) = 1 - \sum_{N=0}^{N=N_{MAX}} \frac{\lambda^N e^{-\lambda}}{N!}$$

When dividing into segments, the probability of not missing a hit is equal to the probability of not missing any hit in any segment. With N segments in a line, the probability of missing a hit in a line therefore becomes:

$$\begin{aligned} P(missing\ hits\ in\ a\ line) &= 1 - P(not\ missing\ hits\ in\ a\ line) \\ &= 1 - (P(not\ missing\ hits\ in\ a\ segment)^N) \\ &= 1 - ((1 - P(missing\ hits\ in\ a\ segment))^N) \end{aligned}$$

One here sees that for an increasing number of segments, the probability of missing hits within a segment must decrease if the probability of missing hits within the entire line is to be kept constant.

Depending on the physics experiment, there will be different hit densities, and therefore different requirements on the number of hits per line that have to be taken into account.

To perform the segmentation, one therefore has to start with the average number of hits per chip and divide by the number of rows to obtain the average number of hits per line. The average number of hits per line then has to be divided by the number of assumed segments, and the cumulative Poisson distribution function must be used to find the probability of missing a hit within the segment. Having the probability of missing a hit within a segment, the formula above can be used to verify if the probability of missing hits within each segment is low enough to keep the probability of missing hits within a line below the acceptable level.

As a design rule of thumb, a probability of 10^{-3} for missing a hit in one line has been considered acceptable for the development of sensors in the MIMOSA line of chips. With a probility of missing hits below 10^{-3}, the fraction of lost hits is below 0.1%, and therefore at the same order as the error in MIP detection efficiency measurements (personal correspondance with Rita De Masi).

The same considerations apply for the clusterization circuitry, although the cluster finder is capable of encoding every hit as one cluster, and therefore can process matrices with densities up to six hits per line segment without missing hits.

2.5 Line address suppression and other data formatting issues

2.5.1 Line address suppression

To reconstruct the hit patterns of the pixel matrix, it is necessary to provide the encoded states with both their row address and their column address.

If the data is stored and transferred as 16-bit words, and if 10 address bits are assumed for both line and column, then each state must be encoded with two words ($10 + 10 + 2 = 22$ bits, fitting into two 16-bit words), and one hit therefore becomes $2 \times 3 = 6$ words.

However, if a data format was introduced where a 16-bit line address header precedes all the column addressed states in the line, then the line addresses of the states succeding the first state in the line can be suppressed. This line address suppression would be efficient in terms of data reduction as soon as the number of states in a line exceeds one. As will be shown in Chapter 3, a data format with line address suppression has been introduced for the state encoding circuitry of MIMOSA26.

2.5.2 Other data formatting issues

Inside the sensor readout electronics, memory width is not necessarily a problem, as the designer has the freedom to compile memories of custom width. But if a standard communication protocol is to be used for transferring the encoded states to the external data acquisition system, it is normally desirable that the transferred packages are multiples of 8. For example, with the 8b10b series transmission protocol that is to be further described in Chapter 4, the data is read as 8-bit words before being expanded to 10 bits for providing an embedded clock in the series transmission. If an encoded cluster is 18 bits wide (10 column address bits and 8 bits for encoding), it must therefore most likely be transmitted as 24 bits or 32 bits, and hence, bandwidth is wasted.

An unconventional word width is also inconvenient with regard to storage on the receiver side. As long as ASIC circuits are utilized, the ASIC designer has the full freedom of utilising whatever memory width that is desired, but if the data is to be stored in the memory of a data acquisition system based on industrial standards, it has to comply with a standard memory size. If an 18-bit encoded cluster has to be stored in the computer memory as a 32-bit word, while the corresponding state encoded cluster can be stored as three 16-bit words, the reduction in memory storage for encoded clusters compared to encoded states will only be 33 %, instead of 50 % as otherwise could have been the case.

With each 18-bit cluster having to fit into a 32-bit word, there would still be 14 unused bits for each encoded cluster, and therefore in the end sufficient space to not only store the column address, but also the entire line address. The line address suppression used in state encoding would therefore be of little use with clusterization, as it would in fact only increase the amount of data to transfer.

However, the overhead due to formatting into standard word sizes also gives the advantage of having ”‘reserved bits”’ that are useful for the scalability of the system, for example if the column address range of the pixel matrix is to be expanded.

2.6 On-chip buffering and memory issues

If the pixel matrix is processed line by line, the extracted states will arrive at an uneven, Poisson-distributed rate, ranging from nine states in the worst-case lines to zero states in the best-case lines. To keep the number of transmission links at a minimum, and with a data rate that minimizes the load on the data acquisition system, it is therefore desired to provide some kind of buffering. The lowest number of serial links, alternatively the lowest allowable bandwidth, is obtained when the buffer is dimensioned in such a way that the number of words transferred during one frame comes close to the average number of states identified during one frame, including their line addresses.

If a FIFO is utilized as a buffer between the state selecting multiplexer and the serial transmitter, it can according to queue theory be modeled as a queue of type M/D/1 (Kendall’s notation), where the arrival of service requests is Poisson distributed and the service rate is deterministic (see for example [57]). The size of the FIFO is then to be modeled on the basis of desired service rate and the tolerated probability of blocking the queue. In this way, it is thus possible to arrive at an optimally dimensioned FIFO, where memory size is traded against reduced bandwidth of the serial transmitter.

However, with the MIMOSA26 architecture, which is to be further described in Chapter 3, another approach is followed. A ping-pong buffer was implemented, where one half was being filled up with encoded states by the multiplexer while the other half was being read by the serial transmitter, and the two halves switched at the initiation of each frame.

With such an approach, both the two halves of the ping-pong memory have to be dimensioned for the worst-case of hits per frame, and the lowest data rate of the serial transmitter corresponds to using the entire time interval of one frame to transfer the worst-case amount of encoded states.

The lowest possible data rate is then given as:

$$Minimum\ data\ rate = \frac{Worst\ case\ amount\ of\ encoded\ data}{(Frame\ readout\ time) \cdot (Number\ of\ parallell\ links)}$$

Although this approach means that the memory becomes over-dimensioned with respect to the optimal size, and that the serial transmission data rate must correspond with the worst-case amount of hits instead of being relaxed with respect to the average amount of hits, it also provides more robustness and scalability. Even the most unthinkable scenarios where all the worst-case amounts of hits in one frame arrive at its very end, and all the worst-case amounts of hits in the next frame arrive at its very beginning, will be handled by this buffer without being overflowed.

If the hit density is large and uniform enough to give a minimum of one state in each line, then the total number of words for an image would be the total number of rows plus the maximally expected number of states per chip per frame. For hits due to particles, the number of hits must be multiplied by three to take into account that there are three states for each regular cluster. For hits due to noisy pixels, one can assume one state per fired pixel:

$$Memory\ size = (Number\ of\ rows) + (hits/chip/frame) \cdot 3 + pixels/chip \cdot (fake\ hit\ rate)$$

For the MIMOSA26, that is targeted for EUDET JRA-1 and utilizing state encoding, the average number of hits per chip is 100, which using the Poisson distribution increases to 132 hits for obtaining a probability of missing hits below 10^{-3}. Also assuming a fake hit rate of 10^{-4}, the required memory size of the ping pong buffer becomes:

$$Memory\ size = 576 + 132 \cdot 3 + 576 \cdot 1152 \cdot 10^{-4} = 1038.$$

As will be further described in the next chapter, the ping pong buffer of MIMOSA26 is composed by 2 x 2 memory banks of 600 words, which according to the calculations should be sufficient for EUDET JRA-1.

2.7 Conclusion

Based on the physical model of clusterized hit patterns in MAPS technology, and assuming experiments with sufficiently low occupancies ($< 1\%$), zero suppression algorithms and circuits were developed to reduce sensor data rates with more than an order of magnitude, and to reduce the load on the data acquisition by performing the first steps of track reconstruction on-chip. In the next chapter, we will present the MIMOSA26, which is the first sensor in the MIMOSA series of chips that embeds such functionality.

Chapter 3

MIMOSA26 - Sensor with data sparsification

In the autumn of 2008, a new version of the MIMOSA series of chips, the MIMOSA26, was developed and submitted. The MIMOSA26 was designed to equip the EUDET JRA-1 beam telescope [2], but with zero-suppression circuitry dimensioned for the STAR experiment, where 200 hits per chip were originally assumed (STAR 200), and with charactistics given in Table 1.1 of Chapter 1. EUDET JRA-1 is part of the research and development program towards the international linear collider.

The MAPS technology limits the logic of the pixel cells both in terms of space and in terms of transistors (only NMOS allowed), and this makes it difficult to implement sparsified readout architectures similar to those of hybrid detectors. However, by activating the in-pixel preamplifers row by row (rolling shutter), the pixel signals can be fed line-wise to the periphery of the matrix for discrimination, data sparsification, buffering, and serial transmission.

In the MIMOSA series of sensors, MIMOSA26 is the first readout architecture to include sparsified readout and serialization. The data sparsification is based on the state encoding algorithm described in Chapter 2.

The concept of utilising sparse scan for column-wise identification of hits has already been implemented in the bump bonded ATLAS pixel frontend [50]. However, the combination of operating the matrix in a rolling shutter mode, performing line-wise hit identification through a sparse scan algorithm, and to combine the sparse scan with an addition stage of data sparsification through state encoding, is a new feature that has never before been implemented.

The pixel matrix of MIMOSA26 consists of 1152 columns and 576 rows. All the 576 rows are multiplexed onto one single line of 1152 binary discrimators (one for each column) and the discriminated output of each line is state encoded. A multiplexer then selects the encoded states and stores them into an embedded memory. A serializer reads the memory and transfers the data on an an LVDS link.

The pixel pitch of the MIMOSA26 is 18.4 µm, and with 576 rows, the total height of the pixel matrix becomes approximately 10 mm. The peripheral readout electronics, with discriminators, SDS, multiplexers, memory management and serial transmission, has a total height of 2.5 mm. Of these 2.5 mm, 400 µm are due to the discriminators and 310 µm are due to the zero suppression circuitry (sparse scan with state encoding).

3.1 Achieved performances of MIMOSA series of pixel detectors

3.1.1 Background

Up to now, more than 20 different MIMOSA sensors have been designed, fabricated and tested, in 7 different manufacturing processes, with the AMS-0.35 opto process proving to be the most satisfactory technology.

The first prototype chip to implement column-level discrimination was MIMOSA-8. Fabricated in TSMC-0.25 μm techology, it featured 32 columns with 128 pixels of pitch 25 μm. The columns were read out in parallell, with 24 columns equipped with integrated discriminators and the remaining 8 columns delivering an analog output signal used for test purposes. A pedestal subtraction was performed before discriminating the signals with the help of correlated double sampling (CDS).

MIMOSA-8 showed good results, but as the TSMC-0.25 only had 7 micrometers of epitaxial layer, compared to the 14 micrometers provided by the AMS-0.35 opto process, it was expected that the pixel signal could be improved by reimplementing the same architecture in AMS. With MIMOSA-16, the MIMOSA-8 was transferred into AMS technology.

Following MIMOSA16, the next steps that had to be taken to obtain a sensor targeted for experiments and applications like STAR and EUDET, was to reduce the pixel pitch from 25 to 18.4 µm, optimize the diode sizes (large enough to give sufficient charge collection efficiency and small enough to give sufficient gain). It was also necessary to optimize the discriminators for a smaller pitch and to integrate data compression logic and memories for data storage [58].

3.1.2 The MIMOSA22 and SUZE prototypes

As a first step towards incorporating zero suppression into the MIMOSAs, two prototype circuits were manufactured - one for the analog sensor and binary discrimation part, called MIMOSA22, and another one for the digital part, called SUZE (Suppression of Zeroes). The purpose of manufacturing two circuits was to prove the principles of the analog and digital parts separately, and as a next step to develop a new chip where the state encoding and the zero suppression were integrated together with a real sensor. Both chips were designed and manufactured in the AMSC35B4O1 opto process.

To identify the optimum pixel configuration, with respect to charge collection efficiency, signal-to-noise-ratio, charge to voltage conversion, and post-irradiation noise performance, the MIMOSA22 contained totally 17 different pixel configurations. Results from both beam tests and laboratory tests showed that the optimum pixel configuration was a combination of a radiation-hardened n-well, an enhanced load to improve gain, and feedback to suppress post-irradiation fixed pattern noise.

The zero suppression micro circuit, SUZE, is based on the line-wise state encoding principles described in Chapter 2. At the first input stage, there are two parallell 64-bit banks of sparse-scan with state encoding. Then, at the second stage, a multiplexer selects up to nine extracted states, which at a third stage are stored to a memory until being read out. Although the SUZE circuitry only processes two 64-bit line segments, it is constructed to scale with a larger number of line segments. There is no sensor connected to the SUZE circuitry, and to provide the full freedom of testing all kinds of line patterns, the SUZE prototype is designed to swap between reading two independent registers that each have a length of 128 bits (two banks of 64 pixel inputs). The number of rows to process is determined by a programmable register containing the maximum number of rows. There are also four registers containing the start and stop addresses for two empty windows in the matrix. In this way, the SUZE sees a virtual pixel matrix with programmable content. The configuration registers and the registers containing the two test lines are loaded through a JTAG slow control unit.

Following successfull SUZE tests in spring 2008, the next step was to integrate the zero suppression circuitry together with an actual pixel matrix.

3.2 Architecture of MIMOSA26

The readout and data sparsification architecture of MIMOSA26 interfaces a pixel matrix composed of 1152 columns and 576 rows. At the bottom of the pixel matrix, peripheral space is allocated for succesively locating discriminators, sparse-scan with state encoding, a multiplexer structure for state selection, a memory management unit with ping pong buffer and a serializer for data transmission. The readout electronics is programmable through a JTAG interface.

Figure 3.1 shows a functional block diagram of the architecture.

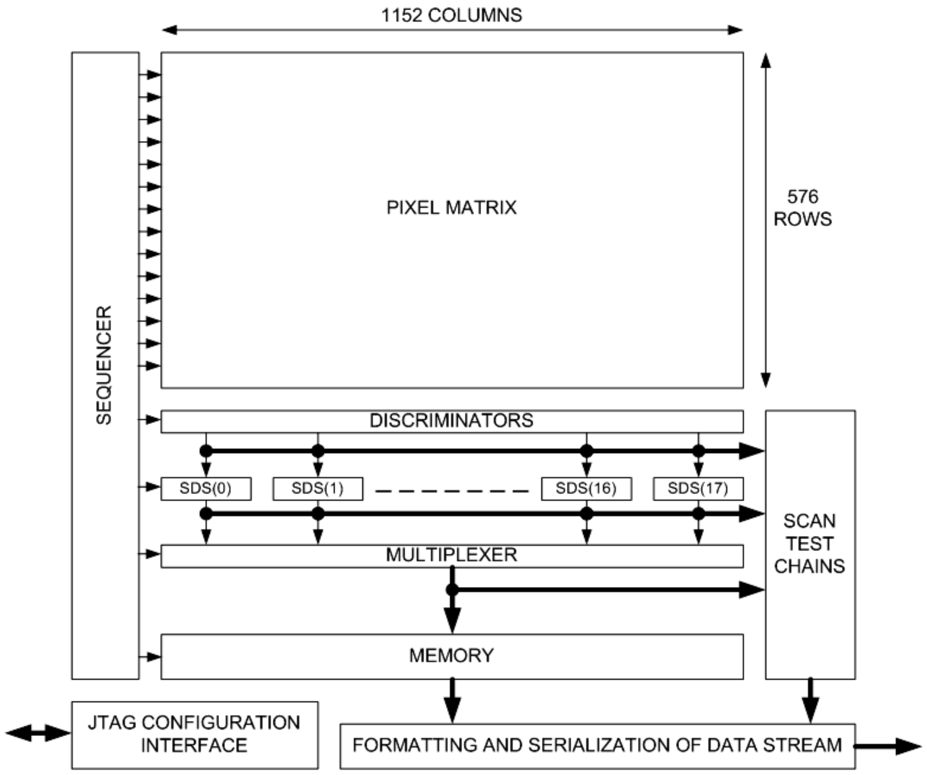


Figure 3.1: Functional view of MIMOSA26.

The modules that are to be presented in further detail are:

- The pixel cells. Each pixel cell contains charge collection diode and a low-gain preamplifier.
- The discriminators. The discriminators perform the single-bit A/D conversion of the analog pixel signals. The threshold of the discriminators is programmable.
- The sequencer. The sequencer activates the lines in the pixel matrix one by one, and it generates all the synchronization signals for the line processing.

With all the synchronization signals being programmable, the sequencer offers a great flexibility for use.

- The sparse scan with state encoding. It extracts the useful data from the raw binary stream of the pixel line, mostly comprised of zeroes, by encoding addressed states that identify and model the patterns of non-zero bits. The SDS circuitry is split into 18 parallell segments, where each segment may extract up to six states.
- The multiplexer. It examines the status of all the 18 segments, and selects up to 9 states.
- The memory management. The memory management buffers the content of the current frame while reading out the content of the previous frame. A serializer reads the memories and formats it into two serial streams.
- The JTAG interface, which provides access to all the configuration registers and the I/O pins of the chip.
- The scan chain. It provides access to the outputs of the individual stages in the data processing pipeline and is implemented for debugging purposes. Since the JTAG boundary scan interface is too slow (< 10 MHz) for reading out the test results of each frame in real-time, the scain chain was implemented as a custom chain that uses the fast LVDS serial link to transmit the sampled test results.

3.2.1 Sensors part

Pixel cells

When a hit arrives, it results in a voltage step in the charge collecting node. When the amplifier is turned on, the same voltage step is amplified and fed through a capacitor to arrive at another node that is buffered by a source follower. The height of the step signal is then sampled and compared with a reference value of the pixel. A functional block diagram for a pixel cell and its interfaces is shown in Figure 3.2.

The sequencer controls the readout of the pixel cells in each row, and the timing waveform is shown in Figure 3.3. The operation of the rows is pipelined to minimalize line processing time. While still processing the current row, the sequencer already powers on the next row by asserting the appropriate bit in the PWR_ON signal array. It then selects the next row by asserting the appropriate bit of the SEL_ROW signal array, and the pixel signal is read by asserting the READ signal. After reading the pixel signal, the source follower is reset to its operating point by

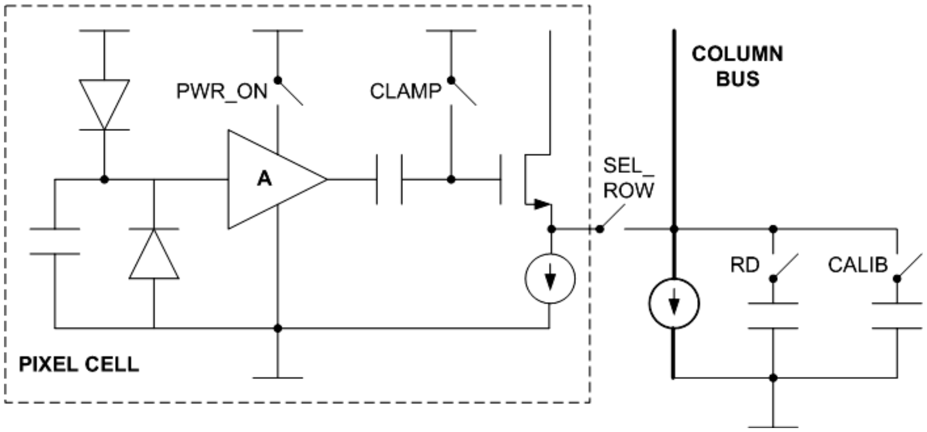


Figure 3.2: Block diagram of pixel cell with interfaces

asserting the RST and CLAMP signals. The reference value is read by asserting the CALIB signal, and the discriminator output is finally latched by asserting the LATCH signal.

The correlated double sampling is performed to suppress the DC-offsets between pixels (fixed pattern noise) and correlated noise sources like common mode [59].

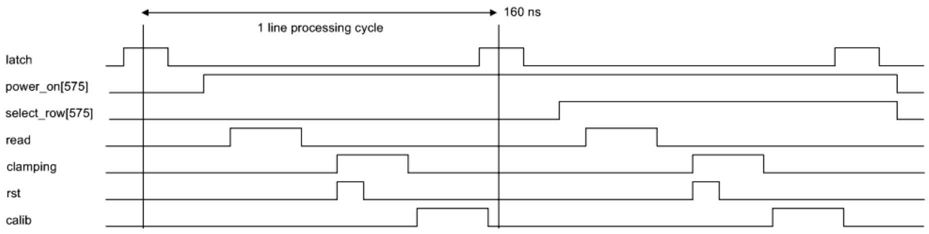


Figure 3.3: Waveform for analog pixel cell readout.

Discriminators

The discriminator compares the differential double sampled signal with a differential threshold voltage, and outputs a binary one if it exceeds the differential threshold.

Figure 3.4 shows its principle of operation, with VREF1 minus VREF2 being the programmable differential threshold:

- In the READ phase, the sampled pixel signal is connected to the positive input of the G_0 amplifier with VREF1 connected to the negative input. At the same

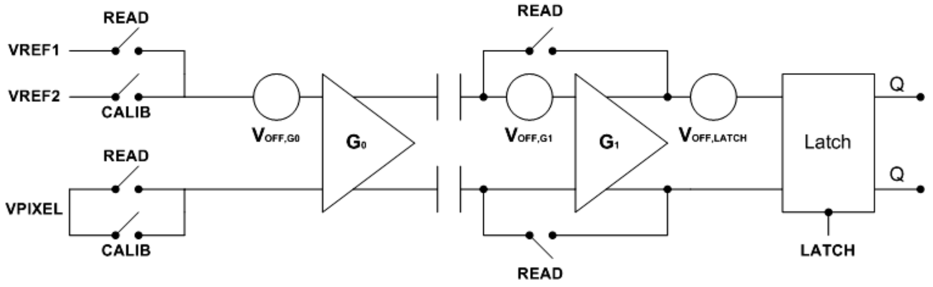


Figure 3.4: Discriminator and correlated double sampling operating principle.

time, the inputs of G_1 are reset to its operating point by closing two switches that establish a unit feedback loop. The differential input of G_1 is capacitively coupled to the differential output of G_0. When the DC values of the amplifiers have settled, all the switches are reopened, and it is time for the next phase, the CALIB phase.

- In the CALIB phase, the sampled pixel reference level is connected to the positive input of the G_0 amplifier, with VREF2 connected to the negative input. As a result, a differential voltage step appears at the G_0 output. The voltage step goes through the capacitors and enters the inputs of G_1. The differential signal is amplified by G_1 and fed into a latch in the LATCH phase that comes at the end of the CALIB phase.

By capacitively coupling the two first stages together, and by establishing a unit feedback loop in the second stage to suppress the initial voltage step in the READ phase, only the potential difference between the the pixel signal and its reference value, after subtracting the differential threshold voltage, is fed to the next stage at the CALIB phase. The major benefit of using this amplifier configuration is therefore that it effectively removes the input offset voltage of the first-stage G_0 preamplifier [60][61].

In case of large amounts of hot pixels in certain columns[1], or malfunctioning discriminators, each individual discriminator can be turned off by setting its corresponding bit in a discriminator disable register.

Test structure for characterizing the discriminators

To obtain the characteristics of the discriminators, it is possible to disconnect them from the pixel column and instead apply a differential test signal, given by the two

[1]Hot pixels are pixels so noisy that they always provide a signal above the threshold of the discriminator.

test voltages VTST1 and VTST2. The VTST1 voltage is applied to the positive discriminator input during the read phase, and the VTST2 voltage is applied during the CALIB phase. Two 8-bit DACs, accessible through the BIAS_GEN register, control the magnitudes of the voltages.

3.2.2 Digital treatment part

Figure 3.5 shows a more detailed view of the MIMOSA26 readout chain and its different stages. The organization of the electronics is similar to the general organization of a pipelined datapath, managed by a controller unit. The controller, called the sequencer, generates all the clocks and control signals required for passing the data stream from the pixel matrix until the serialized output[2]. While the JTAG slow control unit provides all the registers necessary for configuring the sequencer and loading optional test patterns.

The readout electronics can operate in two different modes:

- Normal mode, with the rows of the pixel matrix processed one by one. The sequencer successively selects the individual rows and the discriminators convert selected rows into binary lines before feeding them into the zero suppression circuitry.
- Test mode, where two line registers alternate to provide the readout electronics with programmable test patterns [3]. When operated in test mode, it is also possible to sample the output of each stage in the datapath (SDS outputs or multiplexer outputs). This feature is useful for debugging and identifying the source of eventually detected malfunctions.

Sequencer

To synchronise the different stages of the datapath (both analog and digital), the sequencer generates a number of different clocks and control signals. The most important signals are:

- CKLATCH. The datapath is pipelined and this signal latches the outputs of its different stages.

[2]The main purpose of using a set of different clocks, instead of keeping all of the design synchronous to the system clock, is to reduce the dynamic power consumption. Saving power is necessary both to avoid unnecessary cooling, which comes on the expense of the detector material budget, and to comply with the overall power budget of the detector system.

[3]Since there are only two lines, this configuration does not give the possibility of simulating real clusters which would require three lines of different content. However, since the state encoding is only performed in one dimension, such tests should not be necessary neither.

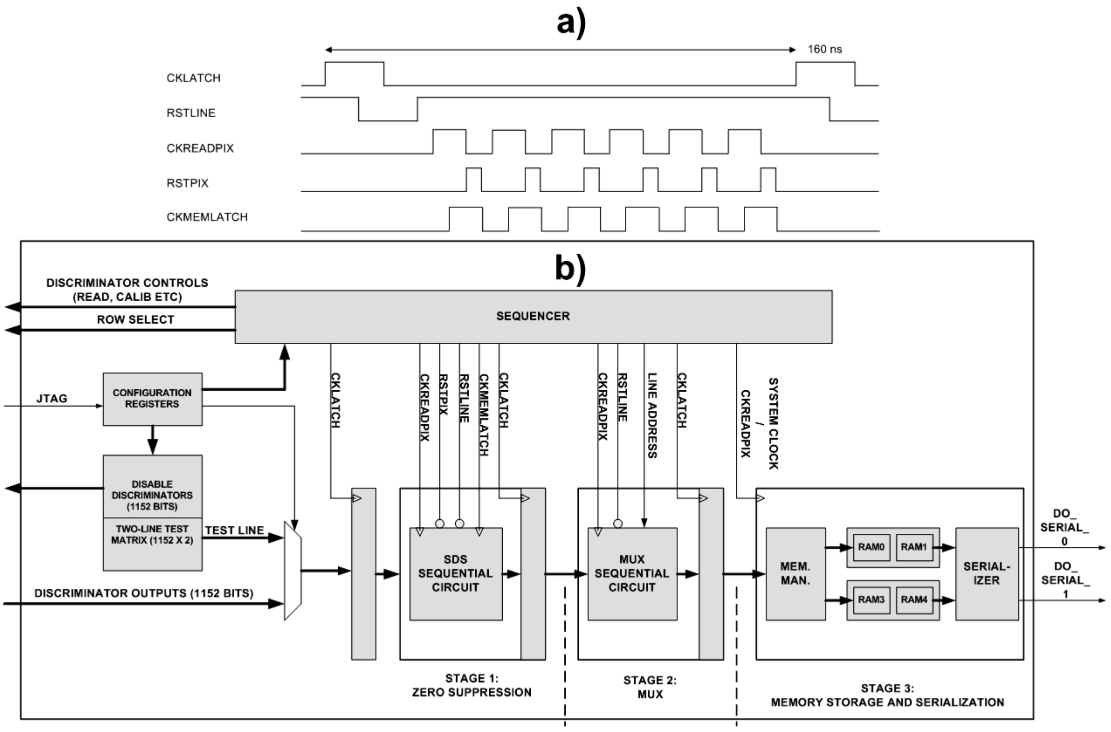


Figure 3.5: Data path with sequencer for the readout and sparsification circuitry. Timing waveform for the control signals is shown in a) and the functional block schematic shown in b).

- CKREADPIX. This signal gives out six pulses for each line. The SDS blocks uses it to extract its six states and the multiplexer uses it to select its nine states.
- READ and CALIB. These two signals are used to control the sampling of the pixels in a row. They belong to the analog part of the datapath.

All the above mentioned signals have the same period of 16 system clock cycles. To make their timing controllable from the outside world, they all map to each their 16-bit register inside the sequencer, where each bit corresponds to one specific time interval in their period. See Figure 3.6.

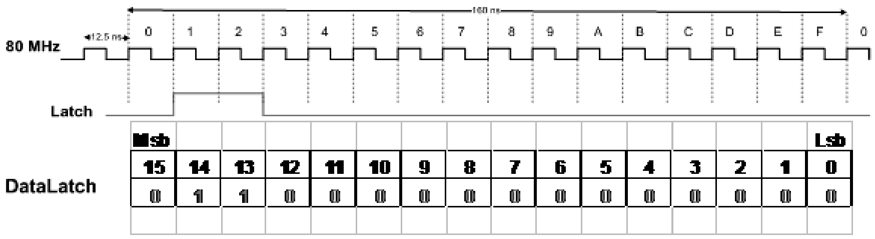


Figure 3.6: The shape of the CKLATCH signal is controlled with a 16-bit signal where each bit correspond to one small time interval in the signal period.

Sparse scan with state encoding

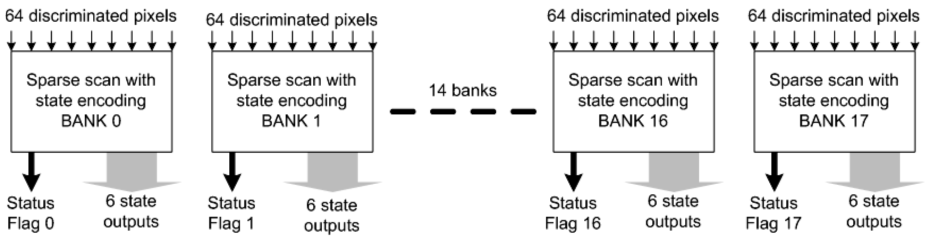


Figure 3.7: The 18 banks of sparse scan with state encoding.

18 separate banks of sparse scan with state encoding interface the outputs of 1152 discriminators. 64 discriminators provide inputs to each bank, and each bank extracts six state outputs and a status flag to indicate the number of valid states. Figure 3.7 shows a functional block diagram of their organization.

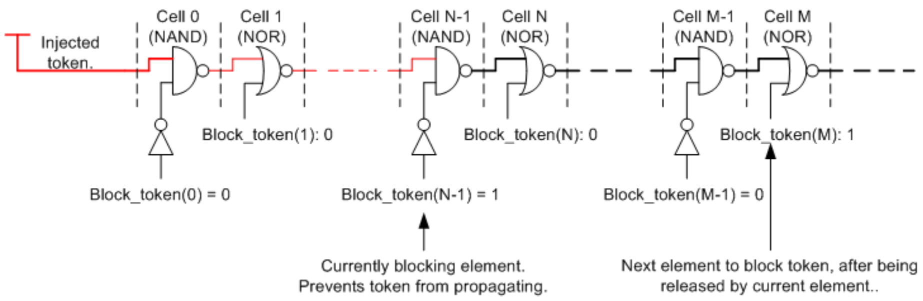


Figure 3.8: Token blocking and token releasing by intertwined NAND/NOR cells in sparsified scan path.

In each of the sparse scan banks, a token is propagated through a chain of intertwined NAND and NOR gates, one gate for each pixel (by mixing NAND and NOR, the propagation delay through the pixel cells is minimized). Each one of the 64 cells in the sparse scan chain are capable of blocking and releasing the token, as shown in Figure 3.8.

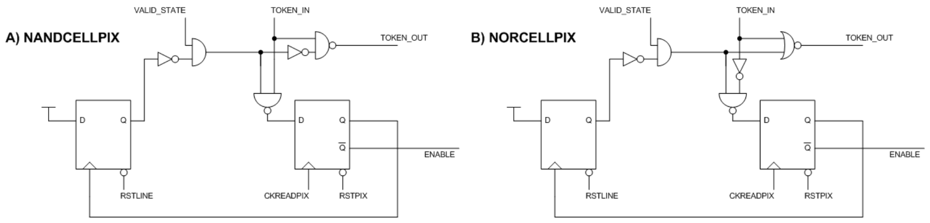


Figure 3.9: Block schematics of NAND and NOR cells in the sparsified scan chain.

The operation principle of the NAND and NOR cells, responsible of blocking and releasing the token, is the following:

For each of the pixel cells, two flip flops are used to control the token passing. The first flip-flop is reset to zero value at the initiation of each line (RSTLINE), and its negative output is AND'ed together with the state validation signal of the column to provide a switch control signal that blocks the token from propagating to the next cell. If there is a valid state, the blocked token is routed into the D input of the second flip-flop, and then clocked into the flip-flop at the first rising edge of CKREADPIX. The negative output of this flip-flop is then used as enable output for the address generation and code selection, while the positive output is used as clock input for the first flip-flop. As the second flip-flop is reset to logical one value with the RSTPIX signal, a rising edge appears at the input of the first flip-flop, a logical one is loaded, and the result is that the waiting token is switched onto the next pixel cell. Figure 3.9 shows the logic of the NAND and NOR cells.

Multiplexer for state selection

The 18 sparsification and state encoding banks are followed by a multiplexer that for each line selects up to nine states. The multiplexer goes through the SDS status flags from the left to the right to identify the banks containing valid states, and the states of these banks are then read and presented to the next output stage.

To meet the timing requirements of 100 MHz operation with 18 banks, the multiplexer splits into two parallell stages, each interfacing nine banks, with outputs merged at a second stage. See Figure 3.10.

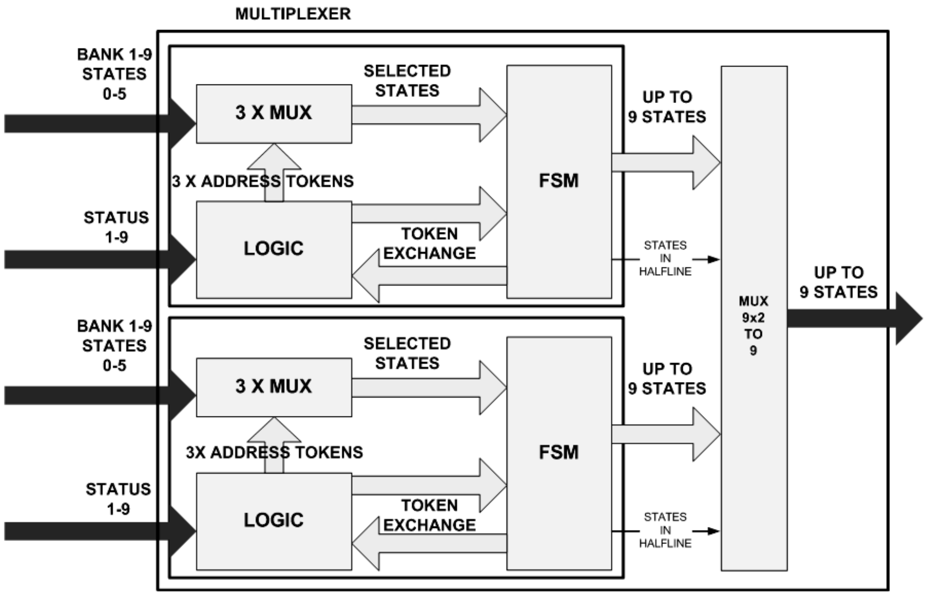


Figure 3.10: Multiplexer with two parallell paths merged in a second stage.

The number of valid states in each bank varies from zero to six. Since the number of clock cycles during one line is limited, it is necessary to have a guaranteed number of selected valid states for each clock cycle, as long as there are valid states left to read. To always have the possibility of selecting three states per clock cycle, three multiplexers point to each their bank, with bank addresses being their select signals.

The multiplexer scans the line from left to the right, starting by selecting the three first banks containing valid states. Then it continues by reading out the valid states of the selected banks. For each time that all valid states are read out of the first of the three selected banks, the multiplexer needs a reference on where to restart the scan for valid banks. This generation of reference bank addresses is implemented as a loop of tokens between two modules.

The first module in the token passing loop is an FSM that performs the two following tasks: 1) It stores into registers the states of the banks currently selected by the three multiplexers. 2) It uses the status information of currently selected banks to request and propose new addresses each time the first selected bank is emptied.

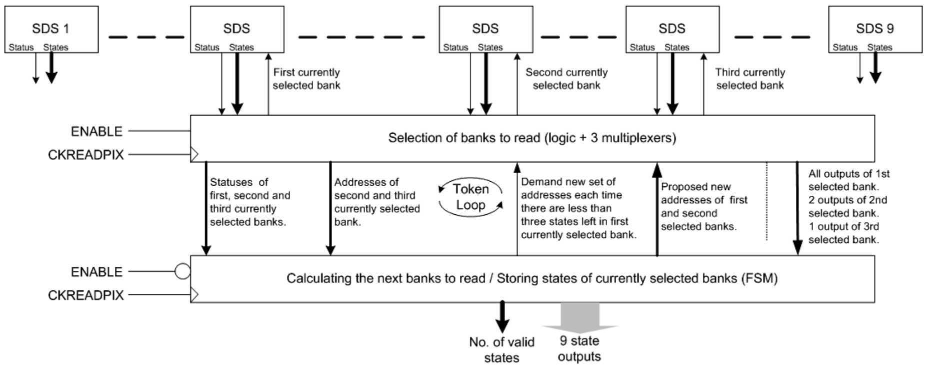

SDS 1
SDS
SDS
SDS
SDS 9
Status
States
First currently selected bank
Second currently selected bank
Third currently selected bank
ENABLE
CKREADPIX
Selection of banks to read (logic + 3 multiplexers)
Statuses of first, second and third currently selected banks.
Addresses of second and third currently selected bank.
Token Loop
Demand new set of addresses each time there are less than three states left in first currently selected bank.
Proposed new addresses of first and second selected banks.
All outputs of 1st selected bank. 2 outputs of 2nd selected bank. 1 output of 3rd selected bank.
Calculating the next banks to read / Storing states of currently selected banks (FSM)
No. of valid states
9 state outputs

Figure 3.11: Functional block diagram of first stage multiplexer with token passing.

The second module is composed of logic that receives the proposed address tokens for performing the final address generation. This module can either accept the two proposed addresses if they point to banks with valid states, and obtain the third address by scanning the status flags of the following banks until arriving at a bank with non-zero status flag. Or it can scan the status flags, starting from the address of first non-accepted token, to obtain new tokens.

Both the two modules are synchronous to the same CKREADPIX clock, and two clock cycles are required for exchanging the tokens. An inverting flipflop provides an active low enable for the FSM and an active high enable for the address token selection logic.

For each clock cycle where the FSM is enabled, it is capable of storing three states, and during the six CKREADPIX clock cycles of one line it is therefore possible to select in total (6/2) x 3 = 9 states. Figure 3.11 shows a functional block diagram of the structure.

For the address proposing FSM, the most straightforward case is when there is only one valid state in the first selected bank, only one valid state in the second selected, and only one valid state in the third selected bank. In this case, three totally new address tokens are needed on the next rising clock edge. For all the other combinations, one or both the proposed new addresses have to overlap the current second and third tokens.

Table 3.1 contains the truth table that was used to implement the final algorithm of the FSM.

Table 3.1: The FSM generates two candidate new address tokens, based on (1) the number of states left to read in the currently selected banks, and (2) the addresses of the second and third selected banks. Modified address tokens are generated as long as there are fewer than four states left to process in the first currently selected bank. Their validity is flagged with an enable signal.

First bank	**Second bank**	**Third bank**	**Enable new token**	**Modified token 1**	**Modified token 2**
1	1	1	1	$Third + 1$	$Third + 2$
1	2	-	1	$Third$	$Third + 1$
1	(≤ 3)	-	1	$Second$	$Third$
2	1	-	1	$Third$	$Third + 1$
2	(≤ 2)	-	1	$Second$	$Third$
3	-	-	1	$Second$	$Third$
ELSE (≤ 4)	-	-	0	-	-

Dimensioning of multiplexer and segmentation of SDS

Targeted for STAR 200, the average number of hits per line becomes $\frac{200\ hits}{576\ lines} = 0.35$ hits/line. For a probability of missing hits that is less than 10^{-3}, 3 hits per line must be taken into account, corresponding to the nine states selected by the multiplexer.

Since the state encoding circuitry only has capability of selecting six states for a line, this means that to meet the requirements, the matrix must be split into segments. If the segment size is set to 64 pixels, it can be shown that the probability of having two hits (six states) in the same line segment is on the order of 10^{-5}, which for 18 segments gives a total probability of missing hits that is below 10^{-3}, and thus acceptable. The 1152 columns are therefore divided into 18 banks of 64 columns.

Line data formatting

For each processed line, the multiplexer provides the next stage in the datapath with a data packet containing a line header and up to nine encoded states with

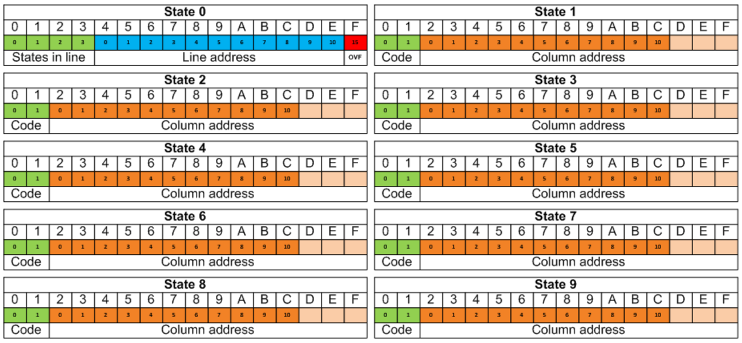


Figure 3.12: Data packet format for a processed line.

their column addresses. The 16-bit line header contains the number of states in the line, the line address and an overflow bit, while the 16-bit states consist of the state encoding and their column address. The full format of the data packet is shown in Figure 3.12.

Memorization

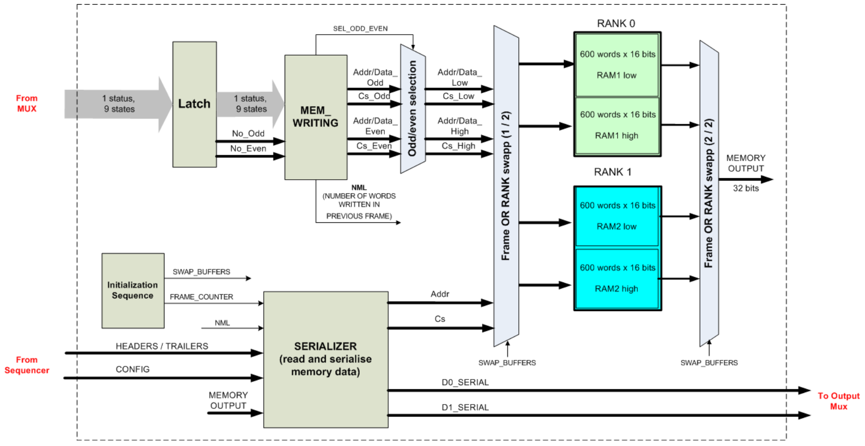


Figure 3.13: Schematic block diagram of memory management.

The third stage in the datapath is the memory management, where the selected states enter a ping pong memory, composed by $2 \cdot 2$ 16-bit single-port RAMs. The two pairs of 16-bit memories switch for each frame that is processed, with the content of the previous frame read out from one pair while the content of the current frame is written to the other pair. The memory management also provides the interface needed to read out the data stored in these memories, initiating the transmission of the previous frame at the start of the next frame.

Figure 3.13 shows a functional block schematic of the memory management.

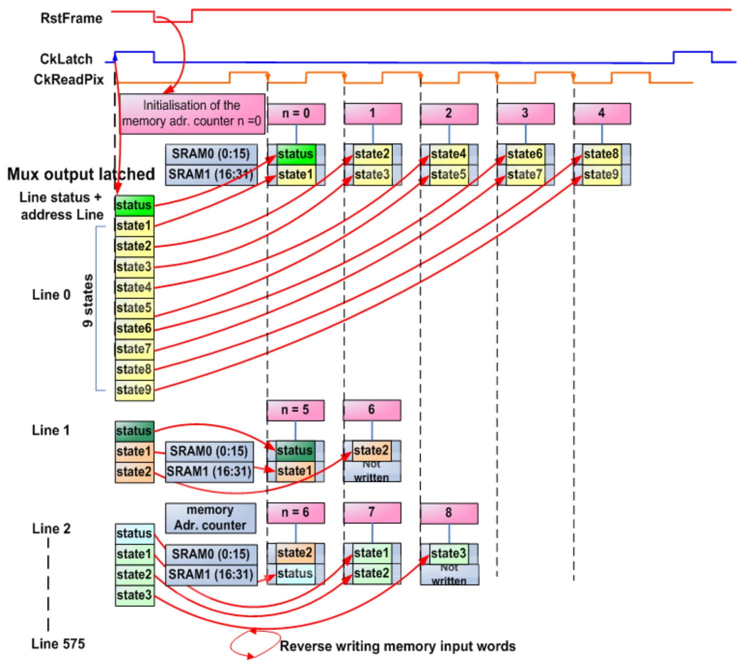


Figure 3.14: Odd/even swapping procedure.

Since the number of selected states varies from line to line, it is not possible, without wasting memory space, to write a fixed amount of words per line to the memories. Instead, the memory interface is able to write both odd and even number of states to the memories, while at the same time keeping track of the total number of states written.

To simplify the writing procedure, the first stage of the memory management unit always initiates the writing procedure by writing in the exactly same order to what it sees as the odd and even memories. While at the same time, it also signals to the odd/even selection logic of the second stage if the data written to the odd and even memories should be swapped or not. And finally, at a third stage, it selects which one of the two memory pairs in the ping pong buffer that is to be written to. The data provided at the input of the memory management is therefore taken through totally two stages of multiplexing before being latched into memory.

The swapping procedure of odd and even words is illustrated in Figure 3.14.

Serial output transmission

In the other end of the memory interface, the two 16-bit memories are read in parallell by a serializer. Basically, the serializer consists of a finite state machine that controls the reading of the two 16-bit memories and the loading of two shift registers. At each their respective state, the serializer loads header, data and trailer information into the shift registers, and subsequently shifts the loaded data out on two LVDS outputs. Figure 3.15 shows a block schematic of the serializer architecture.

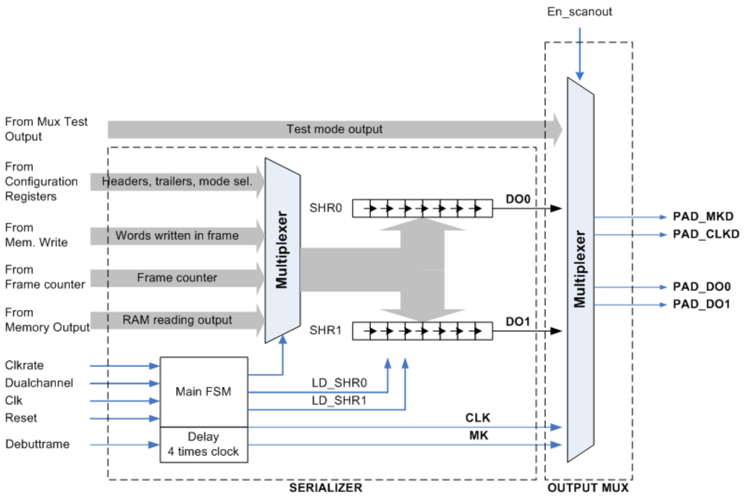


Figure 3.15: Block diagram of serializer and output mux.

The serializer provides in total four external signals: two data signals, one clock signal, and one synchronization strobe. With all four signals implemented as differential pairs, there are in total 2 x 4 = 8 wires for this interface.

The serializer initiates the transmission of each frame by asserting the synchronization strobe while sending the headers, and then a framecounter word follows to indicate the number of data words to be transferred. After transferring all the data words, it ends the transmission by sending a trailer word.

The serializer can operate in four different modes - mode 0, mode 1, mode 2 and mode 3:

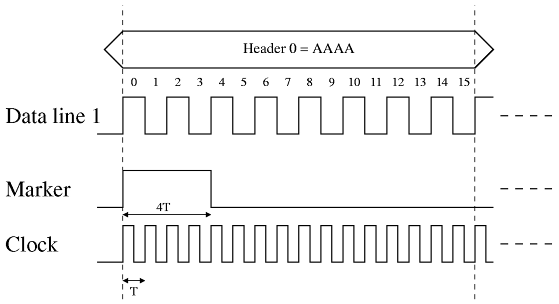


Figure 3.16: Initiation details of serial transfer. When the header of a new frame is sent, a synchronization strobe (marker) is asserted for the first four clock periods of the transmission.

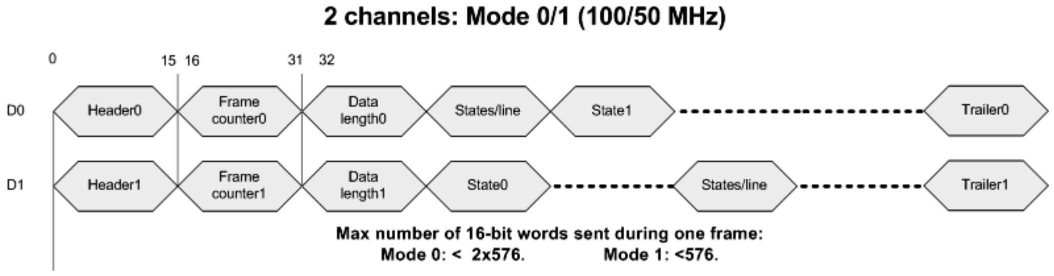


Figure 3.17: Data format of serialization in mode 0 and 1.

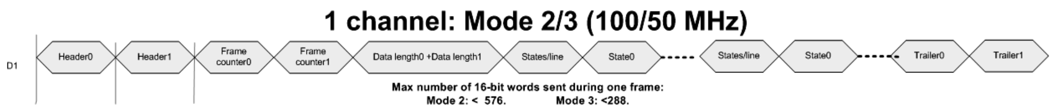


Figure 3.18: Data format of serialization in mode 2 and 3.

- In mode 0 and mode 1, data goes out serially in two parallell LVDS streams, one for each of the two 16-bit memories in the design. The difference between mode 0 and mode 1 is that in mode 0, the LVDS stream is synchronous to a 100 MHz clock, while in mode 1, data are synchronous to a 50 MHz clock.

- In mode 2 and mode 3, data goes out serially in one data stream only, with the content of the two memories being bundled together. All the data is transferred through the first LVDS channel, with the second channel being silent. In the receiving end, the single channel mode of transfer is automatically detected by sensing that HEADER0 and not HEADER1 is the first header to arrive in this second channel. The difference between mode 2 and mode 3 is once again that a 100 MHz clock is used in the second mode and a 50 MHz clock in the third mode.

With the serializer running in mode 0, which is the fastest one, there are in total 576x2 words transferred during one frame (full speed and both links utilized), while a serial transfer with mode 3 can maximally transfer 576/2 = 288 words (half speed and only one link utilized).

Figure 3.16 shows the initiation of a frame in detail. In Figure 3.17, the data format of mode 0 and mode 1 is illustrated. The data format of mode 2 and mode 3 is shown in Figure 3.18.

Communication interface - JTAG

The JTAG slow control unit provides an interface to the outer world for configuration and test. All the configuration registers of the sequencer go through the JTAG interface, and the same applies to the register containing the disable bits for the discriminators and the two line registers that provide inputs to the SDS circuitry when operated in test mode.

JTAG was originally a standard for testing integrated circuits (formalized by IEEE as 1149.1 Standard Test Access Port and Boundary-Scan Architecture), but has also become a standard for programming and configuration of integrated circuits, like microcontrollers and FPGAs. It is a relatively complex communication standard compared to I2C, and compared to newer standards like SPI, it is also a relative slow standard. However, speed is not critical in our application - since the slow control is only used for configuring the circuit prior to normal system operation.

The most important JTAG registers used to control the behavior of the chip are further described in Appendix C.

Scan test chains for discriminators, SDS, and multiplexer outputs

A custom scan chain was inserted for being able to examine the different stages in the datapath and identify possible design failures after the chip has returned from manufacture.

For a selected line, the inputs of a given stage in the pipeline are latched into registers and then the content of these registers are then serially shifted out. Which stage to be shift out is controlled using configuration switches. See Figure 3.19.

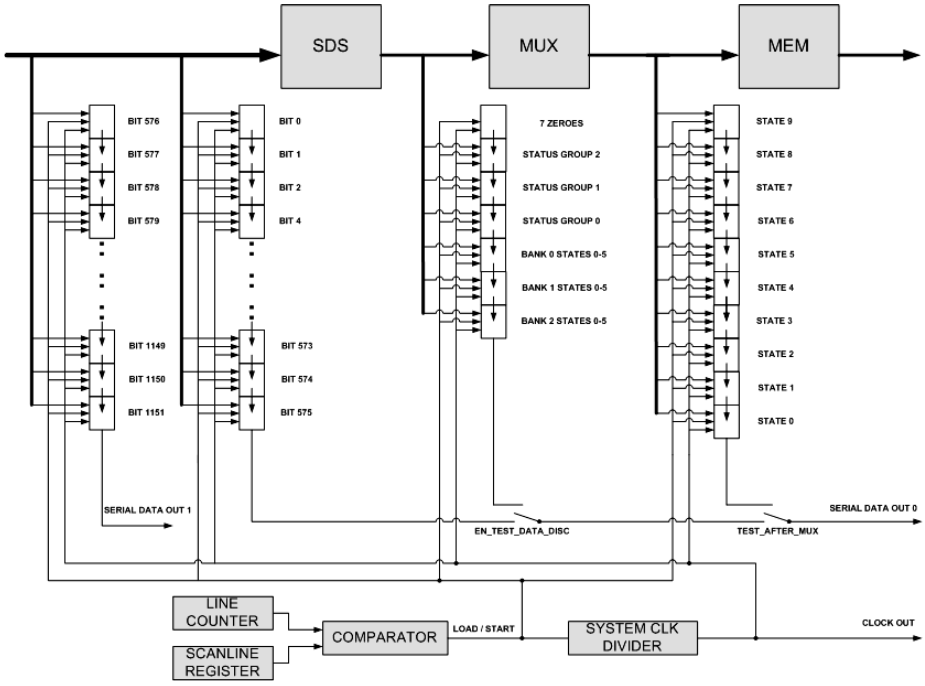


Figure 3.19: Conceptual view of scan test system.

Using the configuration switches, one selects: 1) Which stage in the datapath to test. This can either be the 1152 discriminator outputs, the outputs of the two first SDS banks, or the outputs of the multiplexer. 2) At which line to perform the test. The line to test is either set manually in a configuration register or the line selection is automatized by incrementing the line number for each frame.

A separate configuration register contains all the signals used for controlling the tests:

- EN_SCAN. When this bit is set to one, the scan tests are enabled. Otherwise, the chip runs in normal operation.

- SCANLINE. This register selects which line to check. As soon as we have come to the specified line, the content to be analyzed is sampled and shifted out of the chip.
- EN_AUTO_SCAN. When this bit is set, the line to be scanned is automatically incremented for each frame that is processed (starting at zero and ending at 576). This feature is implemented for automatically scanning through all the lines in the matrix and thereby check if the discriminators are operating correctly for every row.
- ENTESTDATADISC. This switch is used for either selecting the shift register containing the 1152 sampled discriminator outputs or the outputs of the two first SDS banks (states and status flags).
- TEST_AFTER_MUX. This switch is used for either selecting the outputs of the state selecting mulitplexer or selecting the outputs that are already selected by ENTESTDATADISC.

After being registered in the scan chain, the sampled data is bit by bit shifted out of the chip through the LVDS link. Figure 3.20 shows the format of the transferred scan data.

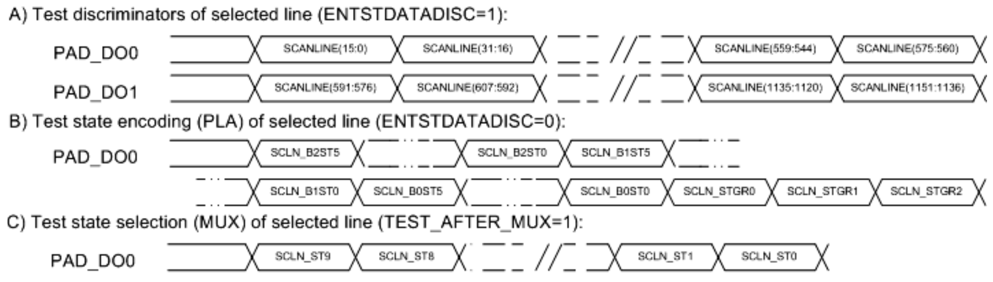


Figure 3.20: Data format for the three different scan tests.

Memory test structures

An optional memory test can be performed in the seven first frames of chip operation. A fixed set of patterns are written into the pingpong memories on the first frame and then read back on the next frame. The data words read back from memory are XOR'ed together with the originally written words, and the bits in the result strings are OR'ed together and routed out to a test pad. The test procedure is described in detail in Appendix D.

3.3 Simulation

For the ASIC development flow, a simulation environment that can provide exhausting testing with all the relevant input combinations is required. It is therefore important to build up an environment of different stimuli and test benches for the modules that make up the ASIC design. There is need for both separate functional tests of all the individual modules and integration tests of the entire system.

When designing VHDL code for hardware, one has to always think carefully about what kind of structures are implied. And normally, only a subset of the VHDL programming language features are utilised - but when designing a testing environment, one also has the full freedom of an ordinary software programmer to create whatever structure is the most convenient. And this is also where the many powerful features of the VHDL programming language, like file handling and polymorphism, come to their right. Writing behavioral algorithms for the testing environment and comparing the outputs of the design with the expected output generated by the test environment is an efficient way to verify the design - since some design failures are hard to uncover before the entire concept is tested.

Within the framework of the MIMOSA26 development, we performed a study concentrated on developing an environment that validates the implemented design against an executable high-level specification. The relevance of the high-level model was verified by first providing input patterns that were assumed relevant for the physical experiment and then successfully reconstructing these patterns by reversing the algorithm of the high-level model.

3.3.1 Simulation environment

The general simulation setup is shown in Figure 3.21. The principle is that the algorithms of the design are implemented twice. First, they are implemented in the simulation environment as high level descriptions of the design, and then they are implemented in the design as state machines. The high level description could be considered to be an executable list of requirements for the design, and if we run the two algorithms in parallell, they should produce the same output. The design is thus validated against the simulation model.

Instead of running the design and the executable specification in parallell, an alternative approach could be to run the design in serial with another module that

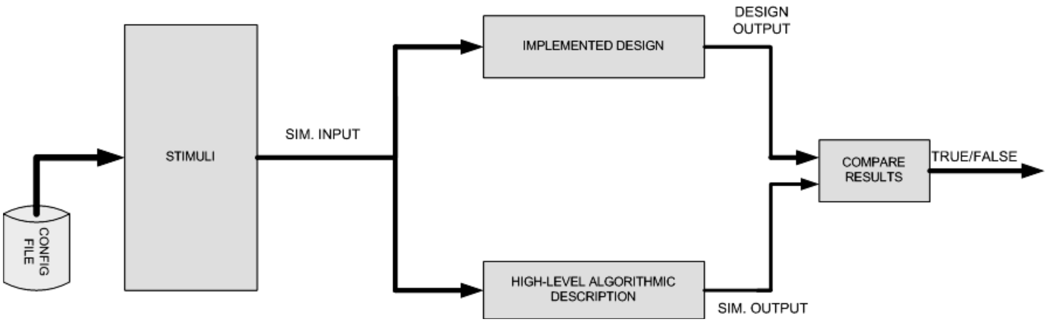


Figure 3.21: General design verification setup.

implemented a reverse algorithm for the signal processing performed in the design, to confirm that the output of the reverse algorithm was equal to the input of the design - and thus confirm that no information is lost. Such an approach is only possible if the processing of data is completely loss-free, and this is not the case in our design where overflows (more than six states in a bank or more than nine states on a line) are simply ignored after flagging an overflow bit. However, such a simulation is still useful to verify that the simulation model and the design is relevant for the most realistic input patterns.

Two simulation environments are developed. In the first simulation, the datapath of the design is tested directly with different kinds of stimuli. In the second simulation, the entire design is stimulated from the outside through the JTAG interface. While the first setup gives the opportunity of performing a much more thoroughly test of all the core parts of the design, the second setup is necessary for testing the design as a black box.

In both simulation scenarios, most of the variable parameters that we feed into the simulation is given through text files that are read by the surrounding test bench entities. This is done with the purpose of speeding up the simulation process. Instead of recompiling the test bench for every small test parameter that is changed, we simply edit the text files and reinvoke the simulator.

3.3.2 Checking the datapath

In Figure 3.22, we can see that the digital datapath, containing sparse scan with state encoding, multiplexer and memory management with serialization, is tested in a standalone test bench environment with inputs given by a stimuli module that reads data from a text file. The purpose of this setup is to validate every single stage of the datapath.

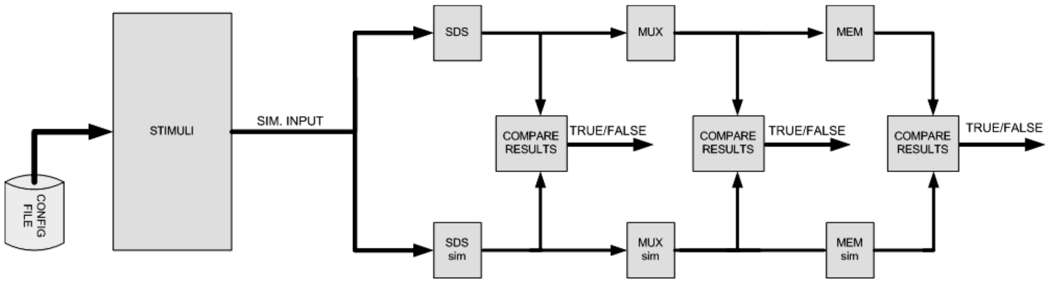


Figure 3.22: Standalone datapath verification setup.

In our test setup, we check in parallell that all the extracted 6 x 18 states are correct, that the nine selected states are correct, and finally that the content on the memories, that are read out through the LVDS links, are also similar to the high-level algorithmic result.

With such a setup, the datapath is simulated standalone with all the required inputs fed from the simulation environment. And with algorithms which at the highest possible level read the inputs, calculate the outputs and compare the results with the outputs produced by the design. In this way, our design datapath is validated at an algorithmic level. As long as the high-level algorithm meets the functional requirements, so must the design also do. Each output stage in the datapath is read and compared against the outputs of the corresponding simulation model, and a flag is asserted to indicate if the results are equal or not.

3.3.3 Checking the entire design

In Figure 3.23, the entire chip is simulated and all stimuli enters from the outside world via the JTAG interface.

In fact, at the top level of the simulation, the testbench is split into two separate blocks, one being the implemented design and another module, called STIM_CORE_DIGITAL, containing all other functions that make up the simulation environment. See Figure 3.24.

The modules that constitute the simulation environment around the implemented design are illustrated in Figure 3.25. They are the following:

- STIMHARDCLOCKS gives out the 100MHz clock of the simulation.

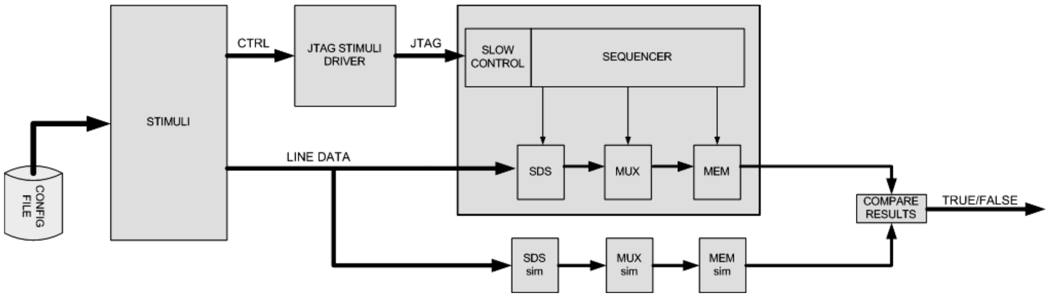


Figure 3.23: Verification setup for entire design with JTAG.

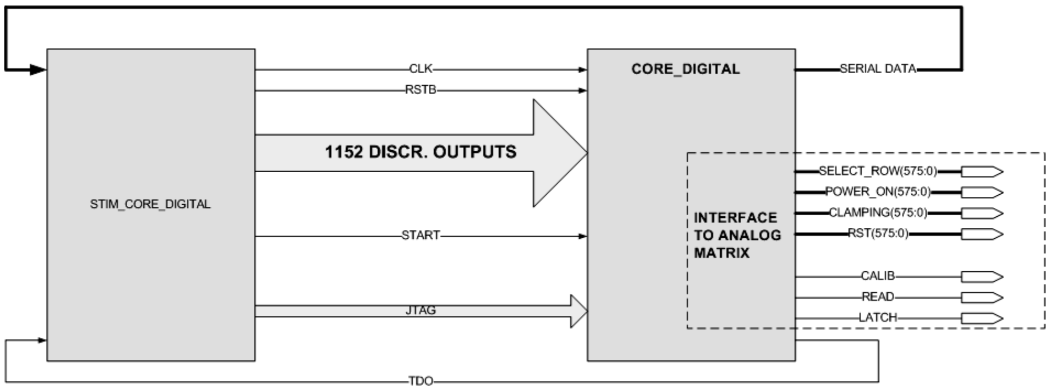


Figure 3.24: Top level of test bench for entire design with JTAG. STIM_CORE_DIGITAL gives all the stimuli and captures all the responses of the design.

- STIMJTAGDRIVER. A module written in Verilog that implements the JTAG protocol. It is used for receiving registers in parallell and feeding these registers into the design through JTAG.
- STIMTOKEN. The simulation setup has an extensive use of text files to communicate between the different modules, and if different users are to run the simulation at the same time, there is need to use different set of text files, one for each user. Each user must therefore have his own instantiation of the test setup, and the token is used to distinguish between the different users.
- STIMREGISTERSCFG. Reads all the simulation configuration parameters from a text file. What kind of text file that is to be read depends on the user token that is connected. The different parameters are read from a text file and delivered to the rest of the design as output signals. The signals that represents JTAG mapped signals are long chains of many differents internal registers. To keep overview of these individual registers, they are internally separated into each their alias.
- STIMSYNCDESIGN. Acts like a supervisor or synchronizer for the stimuli modules and keeps track on where we are in the simulation progress.
- STIMSEQ. Commands the design to start operating after receiving confirmation from the JTAG driver that all the configuration registers are loaded.
- STIMGENPATTERN_FULL_MATRIX. Writes a new pixel matrix to a text file for each new frame.
- STIMGENPATTERN_2_LINES. Generates two lines of hit patterns and feeds them into STIMJTAGDRIVER. It also generates a matrix file with the two-line patterns.
- STIMSIMULIMAGE. Reads the text files containing the pixel matrices to provide the design with discriminator outputs for the current frame. It also gives out the processed data of the previous frame.
- STIMRESULTMEMORY. Compares received serial data from the design with processed data from simulation model.
- PATTERN_RECONSTRUCTOR. Reconstructs an image based on received serial data.

With this setup, it is possible to make a full simulation where the chip is powered on, programmed via JTAG, and then left on its own for running with the loaded parameters.

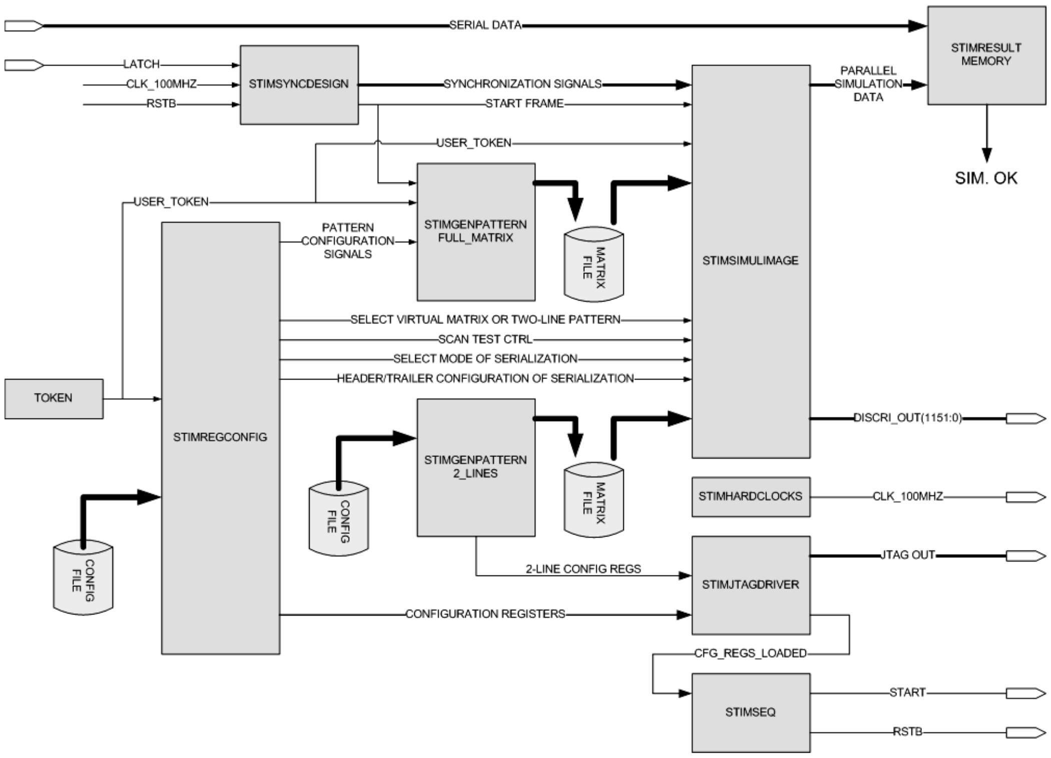


Figure 3.25: Content of stimuli module (STIM_CORE_DIGITAL).

To make the interfaces of the JTAG driver as simple as possible, the STIMJTAG-DRIVER is only fed with five registers (sequencer register, control register, header- and trailer register, and two 1152 bits wide line pattern registers). The five register signals, arriving from STIMREGISTERSCFG, are split into a number of different aliases, and the content of each alias is set by line by line reading a configuration text file. The simulation can be run in two different modes: s

- Normal mode, with the design and the simulation model processing line by line the content of a virtual matrix, generated by STIMGENPATTERN_FULL_MATRIX.
- Two-line test mode, with the design being loaded through JTAG with a two-line pattern.

To be able to test the design in two-line mode with different kinds of two-line patterns, the simulation has been split into simulation cycles.

A simulation cycle starts with the loading of parameters through JTAG. When the JTAG registers are loaded and the design is ready, the design is commanded to start processing data. The data processing then continues for a number of frames specified in the simulation configuration file. When the specified number of frames is reached, the circuit is reset and the loading of parameters through JTAG restarts, but this time with new content in the two line registers. Figure 3.26 shows its principle of operation.

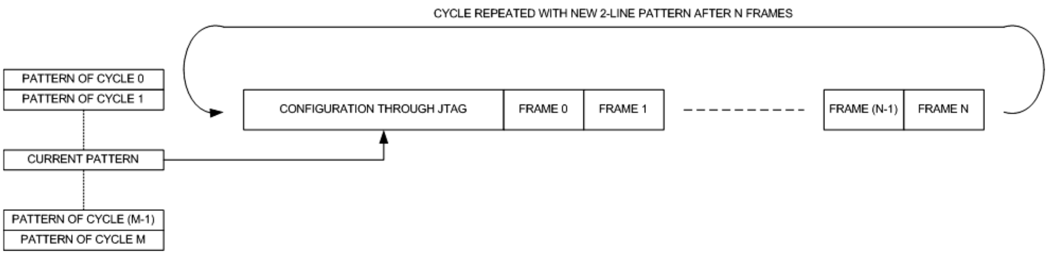


Figure 3.26: Simulation cycle for two-line pattern.

To be able to compare the outputs of the design and the simulation model, the two-line pattern is also written to a text file as a virtual matrix processed by the simulation model.

3.3.4 Input patterns

Due to all the possible configurations of MIMOSA26, there is an overwhelming amount of different combinations to test. A complete back-annotated timing simulation of all the different configuration options is therefore impractical to do (since it will take too long simulation time).

However, it is normally not necessary to perform a full test of all the different combinations if a reasonable test strategy is developed (like in the case of testing manufacturing problems, where almost all manufacture faults will manifest themselves at individual nodes as single-stuck-at-one or single-stuck-at-zero errors). A strategy was developed to create a set of test that would identify as many potential failures as possible.

With all the different test being run with successfull result, the chip is ready for submission.

Unitary tests for all functions

The failures that we would like to uncover were the following:

- Problems with the state encoding of discrimated inputs. For example with clusters overlapping between banks.
- Problems with the selection of states from the individual banks. For example with one state in bank zero and another state in bank nine, activating the critical path in the state selecting multiplexer.
- Problems with the memory management and the serializer. Does it work in all the four different modes?
- Problems with the scan test circuitry.

An assumption was made that whatever combination of states that were selected by the multiplexer, this would not affect the behavior of the serial transmission circuitry (since it simply reads and transfers the states that are stored in memory, without further data processing). The different configurations of the serializer could therefore be tested with a fixed set of discrimator inputs. Similarly, it could also be assumed that the scan test circuitry would function in the same way, regardless of input patterns (as it is simply a scan chain inserted between the pipeline stages to capture whatever content that should be there), and that this circuitry therefore could be tested with a small set of test inputs.

Three types of tests were therefore done for the digital circuitry:

- Testing of the SDS, MUX and memory management for different combinations of discriminator inputs.
- Testing of the serializer for all the different serialization configurations.

- Testing of the scan test circuitry.

A variety of different pattern generating algorithms were developed:

- Patterns that activate the critical paths of the design, especially for the SDS. The signals processed by the SDS have to go all the way through a daisy-chain, much in the same way as a ripple carry adder. A way to active this path is therefore to have one bit in the lower end of each bank and another bit in the upper end of each bank, while leaving the rest of the bits blank.
- Patterns that test the overlapping between banks for all different combinations of states and for all degrees of overlaps.
- Patterns that test if the memory management is working properly for all numbers of states per line and for all total number of states per frame. For this test, the pattern generator started with one and only one hit pixel in the first line of the image, then increasing the number of hits with one for every new frame until nine hits were reached (all the hits were separated by 64 columns and thus located as separate states in each their bank). When nine hits were reached in a line, then the same procedure was repeated in the next line, thus slowly filling up the memories word by word for every new frame.

In addition to these automatically generated patterns, there were also three different two-line patterns that were handcrafted for activating as many failures as possible:

- A pattern that actives the critical path of one bank from one end to the other, while at the same time verifying the overlapping between the two halves of the matrix (as the state selecting mux is split into two parallell paths). See Figure D.1 in Appendix D.
- A pattern that tests out the different types of states while at the same time causing an overflow in the left half of the matrix (Bank 0-8). See Figure D.2 in Appendix D.
- A pattern that tests out the different types of states while at the same time causing an overflow in the right half of the matrix (Bank 9-17). See Figure D.3 in Appendix D.

To test the SDS, MUX and memory management for different discriminator inputs, there were done a number of simulations where the programmable pattern generator of the virtual matrix was run with all the different combinations. This included an automatic test where ten frames were run successively for each kind of pattern generation algorithm.

To test the serializer, the test setup was changed from running with virtual matrix to running with the two-line pattern. Four simulations were done, one for each configuration of the serializer.

To test the scan circuitry, ten different simulations were done with the two-line setup to verify all the different configurations of the scan test circuity (three discrimator tests, three SDS tests, three MUX tests and one automatic discriminator test. It was necessary to make three tests for each scan configuration to test the boundary conditions of line addresses 0 and 575 in addition to a randomly chosen line).

Randomly distributed clusters and pseudo-random images

To provide statistical robustness, three more types of input patterns were developed:

- Pseudo-random. All the rows of the matrix are filled up with LFSR pseudo-random content to test all different combinations of states.
- Masked pseudo-random. Only selected columns are filled up with pseudo-random content, since we do not want more than nine states per line (which lead to overflow). Different kinds of masks were applied. In one case, all pixels except the first and last in every bank were masked out. In another case, a window was diagonally scrolling back and forth in the image - or placed at different random locations for every new frame. The size of the window was increasing for every new frame.
- Randomly placed clusters distributed throughout the matrix. This is a more realistic pattern distribution. The locations of the central pixels in the clusters are determined by using random x and y coordinates. Different kinds of cluster shapes have been tried out. The number of clusters could either be specified by the user or it could increase incrementally from one frame to the next one.

The tests should ideally have been run for some days to obtain a complete verification of the chip. But due to the long simulation time and the short time schedule before submission, each of the three types of test was only run for a few hundreds of frames. However, and as we will see later in this chapter, similar pseudo-random tests were performed as part of the hardware test after the chip returned from submission, with a much larger amount of frames, and these tests showed no signs of failures.

3.3.5 Pattern reconstruction

The method of providing simulation model and implemented circuit with the same set of inputs, and then compare their output, is sufficient to ensure that our design corresponds to the simulated high-level description of the design (provided that a realistic and sufficiently large set of input patterns have been given). During development, the simulation model can then be viewed as an executable specification of the design. However, it is also necessary to have a strategy to verify that the simulation model actually meets the specifications of the chip.

One way for further verification of the simulation model could be to make a full reconstruction of the pixel matrix, based on the data that is provided by the design, and that is already shown to be equal with the output of the high-level behavioral description.

However, it is only possible to perform automatic verification by comparing the original and the reconstructed matrix in the cases of few and sparsely distributed hits, with no more than:

- 6 states per bank, to avoid SDS overflow.
- 9 states per line, to avoid multiplexer overflow.
- 576 states per matrix (assuming state in every line and therefore also one word as line address header for each line), to be able to transfer all data stored in memory during one frame.

Automatic verification of the reconstructed pattern is therefore only possible when there is no line whose number of states exceeds nine and where the total number of states on the entire matrix can be kept within two memories of size 576 x 2 bytes. However, this is, on the other hand, exactly what would be the most realistic model of what is expected from the physical experiment, provided that the assumptions made for dimensioning the chip are correct.

By simulating with input patterns realistic for the experiment, and reconstructing the pixel matrix, one can thus verify if the simulation is based on a meaningful model. A test bench module for storing the deserialized design data, decoding the states, and reconstructing the pixel matrix, was therefore implemented. This pattern reconstructor was tested with a number of different input patterns. What was observed was what could be expected:

- The pattern reconstructor reconstructs states from the right to the left in the matrix, and reconstructs correctly all states, including state nine. If there is an overflow on the line, the nine first states are reconstructed from the right to the left, with the rest of the line left blank.

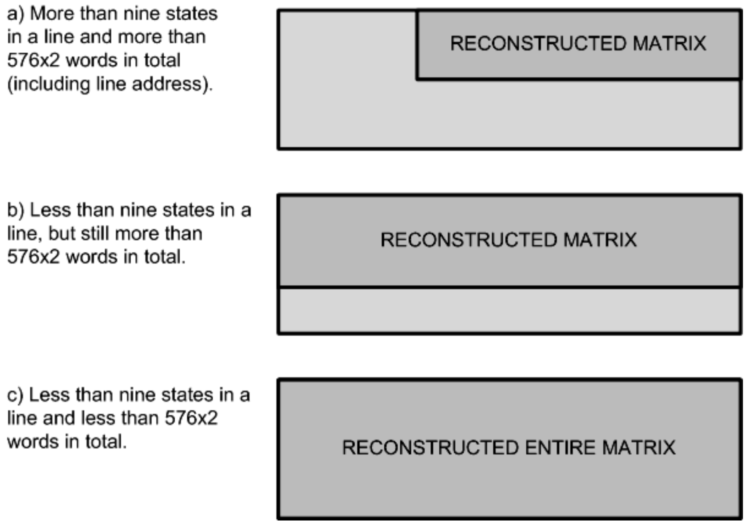

a) More than nine states in a line and more than 576x2 words in total (including line address).
RECONSTRUCTED MATRIX
b) Less than nine states in a line, but still more than 576x2 words in total.
RECONSTRUCTED MATRIX
c) Less than nine states in a line and less than 576x2 words in total.
RECONSTRUCTED ENTIRE MATRIX

Figure 3.27: Reconstructed pixel matrix shown as a function of actual pixel matrix content.

- Dependent on the average number of states in a line, there will also be a maximum number of lines stored in design before the memory is full, with the maximum number of line addresses and encoded states being 576 x 2. If a uniform distribution of states is assumed, with each line requiring a separate line address header, the maximum average of states per line in a correctly reconstructed image becomes one.

Figure 3.27 shows three different cases, with only one giving a fully reconstructed matrix.

After successfully simulated with layout in 100 MHz, and passing all the design rule checks, the design was submitted for manufacture.

3.3.6 Pertinency and weaknesses of simulation

To verify that the design correctly acquires the expected hit information, the design should be tested with input patterns that correspond to the experiment for which the design is targeted. The pattern reconstructor should then be used to reconstruct the image and it should be verified that the only times the hits are missed are when there is overflow, and with a fraction of overflows that corresponds to the accepted probability of missing hits.

The following simulations should therefore have been run:

- Input patterns providing randomly distributed clusters.
- Number of clusters per image varied around the expected number of hits per chip for the given experiment (200 for STAR).
- Physically realistic distribution in cluster shapes, similar to experimental data from beam tests of previous chips.
- For statistical robustness, one needs thousands of frames, and with automatically checking that the only mismatches of the reconstructed patterns are due to overflows.

However, this list of ideal requirements also highlights the weaknesses of the simulation:

- The number of clusters per frame was either fixed or incrementing for every frame, instead of being Poisson distributed around the expectation value.
- The shape of the clusters were in most of the cases purely a regular cluster with multiplicity of 5, instead of being calibrated against experimental data.
- The mismatch comparison of the reconstructed patterns were performed manually, instead of implementing an automatized algorithm for this task, allowing to check a much larger amount of images.
- Only a few hundred images with cluster shaped data were simulated.

The main reason for the lacking features and the small amounts of frames simulated was the tight time schedule before the submission of the chip. While performing beam tests or laboratory tests in real-time, a frame readout time of 200 μs gives 5000 images in just one second. A similar simulation of the 5000 frames would take several days.

3.4 Hardware tests

3.4.1 Test environment

To test the readout and data sparsification electronics of MIMOSA26, an automatized test environment was constructed, with features similar to the simulation environment of MIMOSA26:

- A logic analyser provides stimuli to the design through the JTAG custom interface, and captures its output when running. A C++ test program with graphical interface, running on the logic analyzer, controls the tests.

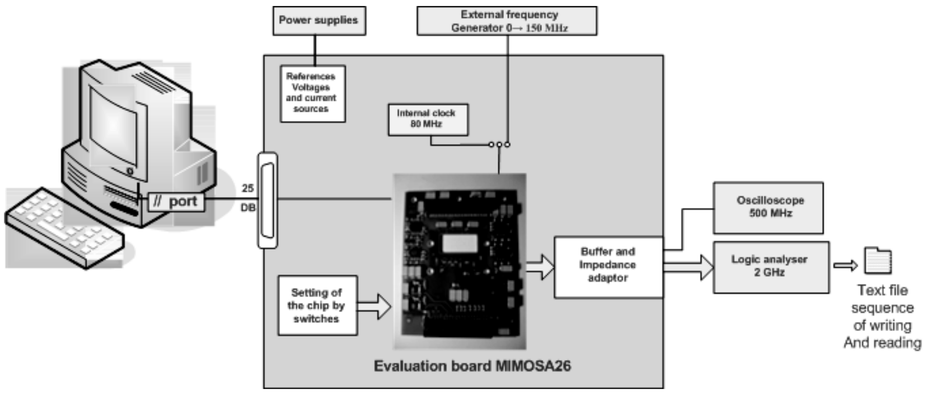

Power supplies
External frequency Generator 0→ 150 MHz
References Voltages and current sources
Internal clock 80 MHz
// port
25 DB
Oscilloscope 500 MHz
Buffer and Impedance adaptor
Logic analyser 2 GHz
Text file sequence of writing And reading
Setting of the chip by switches
Evaluation board MIMOSA26

Figure 3.28: Test setup of MIMOSA26 readout electronics.

- The test program runs in repeating cycles, with each cycle starting with feeding stimuli into MIMOSA26 through the JTAG interface, and then continuing with capturing the outputs of the running system for a specified amount of frames.
- Parallell to the stimuli generation and response capture, the test program also builds a virtual matrix based on the generated stimuli, followed by data processing algorithms that should produce results similar to the captured MIMOSA26 outputs.

The read out responses from MIMOSA26 and the internally generated result could therefore be automatically compared, and a text file with a result report was automatically generated for each frame that was read out. The report contains full information on any detected mismatches between data received from MIMOSA26 and data expected by the test program. The test setup for MIMOSA26 is shown in Figure 3.28.

3.4.2 Performed tests

For the performed tests, one can distinguish between two different types - the qualitative tests and the quantitative tests. In the qualitative tests, the testing program reads data from a directory of files, where each file contains a separate set of manually constructed test patterns - one test pattern for each of the two lines, and eventually one or two windows of empty data. With the quantitative tests, the testing software is no longer based on reading from files. Instead, the testing program generates a pseudo-random test pattern, feeds this test pattern into the design through JTAG and runs one or several frames before the next pseudo-random pattern is loaded and tested out.

According to test theory, even moderate amounts of pseudo-random test vectors are capable of activating a very large amount of potential faults in digital designs (see for example [62]). The optimum test strategy is therefore to test the design by providing input vectors that test the individual functions of the design, and then to use a large quantity of pseudo-random patterns for activating faults not covered in the systematic functionality tests.

A number of systematic tests, quite similar to the simulation tests already performed, were done to the MIMOSA26 chip after returning from the foundry. The systematic functionality tests performed were the following:

- Column address encoding. A single state was subsequently moved from one column to the other to extract the correct column address from all the the different 1152 lines.
- State encoding. The column address encoding test was re-run ten times, ranging from zero to ten states in all column positions of the 18 banks.
- Line address encoding. To verify all the 576 line addresses, three test were performed. In the first test, the chip was loaded with two lines, where each line contained one state. In the two other cases, the first line was loaded with two states while the other line was empty. These two test were each done 1152 times, one for each column.
- Encoding of the shape of states. The column address test was re-run four times, with the consecutive pixels in a state ranging from one to four. This test also verified the continuity between the banks for all the different types of states.
- Overflow detection. Random inputs were provided to the two lines, which in most cases contained more than nine states and therefore should result in an overflow bit being set.
- Serial transmission. Dual-channel and single channel transfers were tested on both 100 MHz and 50 MHz transmission rate.

To provide statistical robustness, two different kinds of tests were also performed:

- Three different patterns, created to activate the critical path through the banks and to verify the overlapping between banks, and to verify that the system correctly overflows, were each tested seven million times;

- 10.000 different pseudo-random patterns were loaded and tested, with each pattern run in 199 consecutive frames.

All these tests were run at 80 MHz operation frequency (in room temperature and with supply voltage of 3.3V). In addition, pseudo-random tests were run with operating frequency ranging from 10 MHz to 115 MHz.

3.5 Architectural improvements with respect to SUZE

To scale with the increase from 128 bits (2 SDS banks) to 1152 bits (18 SDS banks), the original SUZE implementation had to go through some major architectural changes, although its general concepts are still preserved.

The timing critical part of the chip is the sparse scan algorithm and the token generation circuitry of the multiplexer. However, while the SDS chain of the individual banks remains the same, the number of banks that the multiplexer has to go through during one clock period increases from 2 to 18. Such an increase means that the time of the multiplexer for going through each SDS status flag shrinks from $\frac{160\,ns}{2} = 80\,ns$ to $\frac{160\,ns}{18} = 8.8\,ns$.

To meet the timing requirements when the number of banks were increased from 2 to 18, we performed a full redesign of the multiplexer, with the following two major improvements compared to its original structure:

- To reduce the critical combinatorial delay of the multiplexer, the multiplexer structure splits into two parallell paths, each interfacing 9 banks, and with outputs merged at a second stage.

- The clocking scheme of the token passing was changed to make all the multiplexer modules synchronous to the same CKREADPIX clock. Such a reorganization implied that the number of address tokens had to be increased from 2 in SUZE to 3 in MIMOSA26.

Thanks to these improvements, the clock frequency could remain at 100 MHz, and in the same process technology, even after increasing the number of columns from 128 (SUZE) to 1152 (MIMOSA26).

In addition to improving timing, we improved the output data flow by replacing the 32-bit parallell data bus of SUZE with two differential serial interfaces. While parallell buses quickly meet limitations of clock frequencies due to the many signals to interface, differential serial transmission interface can achieve much higher clock frequencies, and they also have much fewer wires, improving material budget.

The serial interface of MIMOSA26 is composed of four signals, two differential data lines, one differential clock line, and one differential strobe line - in total 8 wires. In terms of wires and in terms of achievable frequencies, it is therefore not a fully optimal serial interface, but is the first step towards a serial interface with embedded clock and control.

The splitting into stages and the improved clocking scheme is to be described in further detail:

3.5.1 Parallelization of operations

The state selection multiplexer is composed of two independent instances of the same circuit, whose output results are merged at a second stage. In the first stage, two multiplexer structures, extracting states from each their half of the matrix, are both merging 6 times 9 states into 9 states.

The second stage of the multiplexer, merging the two first parallell stages, is principally another large multiplexer, where the valid states in the upper half of the matrix are combined with the valid states of the lower half, as long as the number of states in the lower half has not already exceeded nine. Its concept is simple: For each of the nine outputs, a multiplexer is implemented. If the number of identified states in the first module, x, is equal to or greater than n, output number n is assigned the nth output of the first module. Otherwise, if x is less than n, output number n is assigned the (n-x)th output of the second module.

To obtain correct addresses for the states in the merged states, the states in the first stage have been assigned bank addresses in the range of 18 to 10, which for the case of the states in the upper half are all being subtracted with nine in the last stage.

With such an implementation, output one would be driven by a multiplexer containing output one from the first module and output one from the second module. While output number nine would be driven by a multiplexer containing output number nine from the first module and all the nine outputs of the second module.

3.5.2 Improved clocking scheme

With SUZE, two address tokens were generated by the multiplexer selection logic at the rising edge of CKREADPIX, and the candidate next tokens were generated by FSM at the falling edge of CKREADPIX. Although this implementation made it possible to generate new tokens at every rising edge of CKREADPIX, it imposes a timing requirement of only 10 ns in the critical path through the multiplexer selection logic.

With the increasing combinatorial delay in the multiplexer selection logic, due to the increasing number of status flags that had to be examined, such a clocking scheme was no longer applicable.

We therefore made the decision to make all the logic sensitive to the rising edge of CKREADPIX, with the multiplexer selection logic providing the address tokens in the first rising edge and the FSM sampling the results and providing candidate next tokens on the next rising edge. An inverting flip flop is used to signal to the two different modules which one is to respond, and with this new scheme it therefore takes two clock cycles instead of one to identify and store a set of selected states. To meet the requirement of identifying nine states on six CKREADPIX cycles, the number of processed states per clock cycle was increased from two to three.

The new clocking scheme improves timing and reduces the problem of clock skew (since a negatively triggered flip-flip in most standard cell libraries synthesizes to a positively triggered flip-flip driven by an inverted clock). A comparison between the new and the old clocking scheme is shown in Figure 3.29.

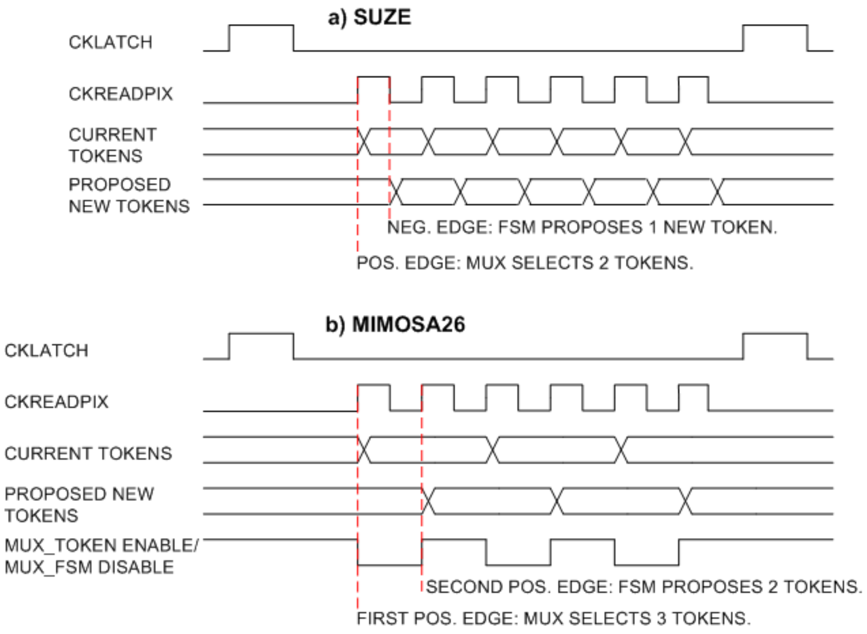

a) SUZE
CKLATCH
CKREADPIX
CURRENT TOKENS
PROPOSED NEW TOKENS
NEG. EDGE: FSM PROPOSES 1 NEW TOKEN.
POS. EDGE: MUX SELECTS 2 TOKENS.
b) MIMOSA26
CKLATCH
CKREADPIX
CURRENT TOKENS
PROPOSED NEW TOKENS
MUX_TOKEN ENABLE/ MUX_FSM DISABLE
SECOND POS. EDGE: FSM PROPOSES 2 TOKENS.
FIRST POS. EDGE: MUX SELECTS 3 TOKENS.

Figure 3.29: a) Token passing handshake of SUZE multiplexer. b) Token passing handshake of MIMOSA26 multiplexer.

3.6 Conclusion

3.6.1 Achieved performances

MIMOSA26 is the first pixel detector with a large-scale matrix (1152 columns x 576 lines), embedded data sparsification, and fast serial transmission. The test results for the digital part shows that the chip is functioning up to an operating frequency of 115 MHz.

The simulation environment being developed together with MIMOSA26 has provided a reference for validating future upgrades of the architecture.

3.6.2 Perspectives of improved timing

Before submitting the ASIC, back annotated timing simulation demonstrated an operating frequency of 100 MHz (worst-case temperature, voltage, and process corner), sufficient for the frame readout time requirements of the EUDET telescope. However, the analysis shows the most timing critical parts of the design and how to improve future versions of the design.

In general, timing can be improved either by changing to a process technology with smaller feature sizes or by improving the architecture. Whatever technology chosen, the architecture should be optimally designed by identifying its critical paths.

In MIMOSA26, three critical paths appear, one in each of the three stages of the pipeline:

- In the sparse scan algorithm, the asynchronous scan of state validation bits arriving from the combinatorial state encoder represents a critical path, with the worst case being one single validation bit on the left side of a 64-bit bank and another single validation bit on the right side. With this scenario, the SDS has only 10 ns (from the assertion of RSTPIX until the rising edge of CKREADPIX) to scan through the entire 64-bit chain of enable bits. Reducing the number of columns per bank improves the SDS timing performance, but for a fixed number of columns per line, a reduction in columns per bank gives a similar increase in banks per line. And increasing number of banks per line increases the critical delay through the state selecting multiplexer of the next stage.

- In the MUX, the scan for non-zero status flags in the individual banks, performed by the multiplexer selection logic, gives a critical path. The worst case is with valid states only in the lower bank and in the upper bank. To meet the

100 MHz requirement, the tasks of the multiplexer are therefore parallellised by splitting it into two branches - one for the nine upper SDS banks and another one for the lower nine SDS banks - with a third module collecting and merging their outputs.

- In the memory management module, the address incrementation of the memories gives a critical path. Since parallellization solves the acute timing problems in the multiplexer, the memory management becomes the part of the design that first stops functioning correctly when the clock frequency is increased.

Optimization of memory interface

The instantiated memories are synchronous to the rising edge of the chip select (CS) signal. However, if we take a closer look at Figure 3.30, we find the signal driving the address input of the memory being sensitive to the falling edge of CKREADPIX [4], the signal driving the data input (the state information) being sensitive to the rising edge of the system clock, and finally the CS input being sensitive to the falling edge of the system clock. The memory is therefore interfaced by three signals arriving from three different clock domains.

As seen from the waveform, the time between the transition of the address signal and the rising edge of CS is nominally 10 ns. However, it is in practice hard to control the timing of signals being sensitive to a falling edge (since the falling edge is synthesized to become the rising edge of an inverted clock), and simulation with back-annotated delay shows that there is less than 10 ns from the transition of the address signal until the rising edge of the CS signal.

A closer look into the logic of the memory management also shows that the data signal presented to the memory is in fact first selected at the rising edge of CKREADPIX, and then delayed with 5 ns by being sampled at the rising edge of the system clock.

However, all information necessary for calculating the addresses and providing the correct states has been available in stable registers since the rising edge of CKLATCH. To improve margins, it is therefore tempting to clear up the clocking scheme and remove all use of falling clock edges, in the same way as already done with the multiplexer.

By making the address generation and data selection sensitive to the rising edge of CKREADPIX, and the CS signal sensitive to the rising edge of the system clock, the timing margins for the address generation increase from 10 ns to 15 ns, and the

[4]With the CKREADPIX of the memory interface, unlike the rest of the design, being delayed with 5 ns and thus sensitive to the falling edge of the system clock.

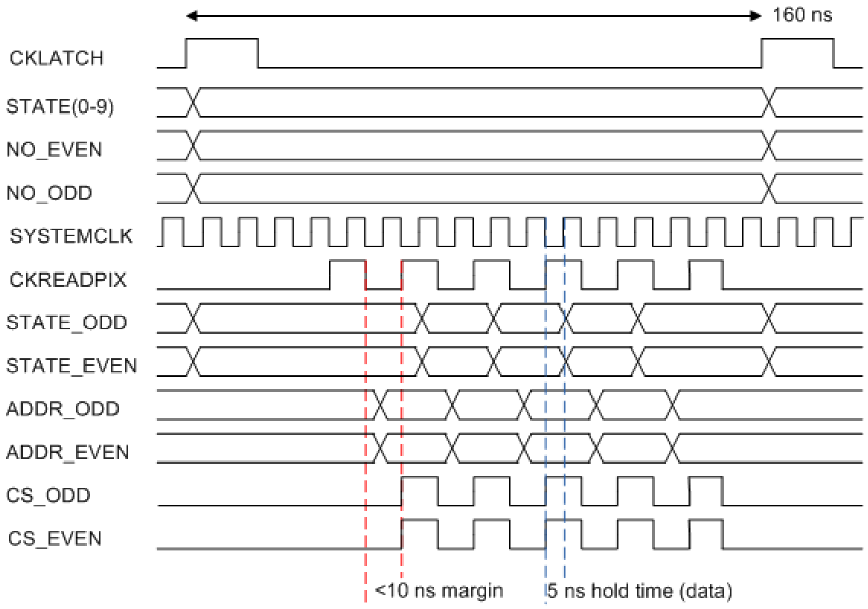


Figure 3.30: Timing waveform for addressing and writing data to odd and even memories.

register for providing 5 ns data delay is avoided completely.

The proposed changes have been written in VHDL and simulated with a large variety of different patterns, all with the same successfull results. The new memory interface is therefore to be used in new versions of MIMOSA chips with zero suppression, like the Ultimate-1 chip that is scheduled for submission in Autumn 2010.

Optimization of sparse scan

The speed of the sparse scan chain can be improved by implementing priority look-ahead logic for the propagating token, in the same way as was done for the ATLAS pixels [50]. In the ATLAS pixels, the lines of pixels are organized into smaller blocks, and at the end of each block, there is also a look-ahead path for bypassing empty block.

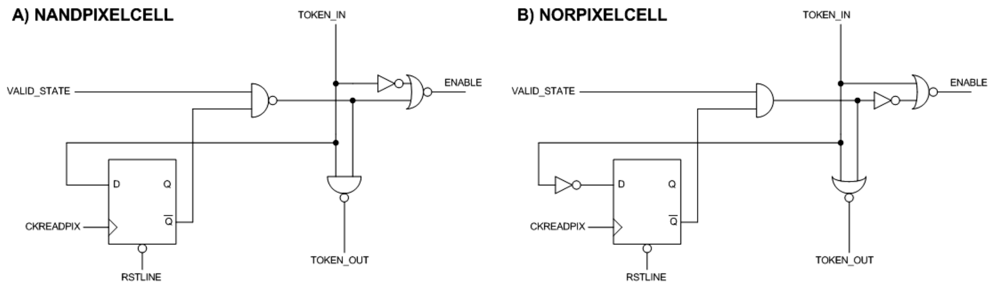


Figure 3.31: Block schematic of improved NAND/NOR cells in sparse scan chain.

Look-ahead techniques are commonly used in adders to improve the speed of signals that have to pass through many elements, like those of the carry generation logic. Instead of having to ripple through large chunks of elements containing zeroes, the elements are organized into larger blocks where logic is implemented to "look ahead" and bypass the empty blocks. With the ATLAS pixels, there is a look-ahead path for each 16th pixel. For the MIMOSA26, however, there is no look-ahead path at all, only one long chain of NAND and NOR gates, and the token has to ripple through all the pixels in each bank. To implement look-ahead logic in the sparse scan chain is therefore one measure that could be taken to improve speed.

The analysis of the pixel elements of the sparse scan also shows that their implementation is suboptimal with respect to logic utilization and timing. Since the token is released from one cell by the RSTPIX signal, and then latched into the logic of the next cell by the CKREADPIX signal, the time for the token to fly through the pixels is limited to 10 ns.

By preserving the NAND/NOR sparse scan chain, but rewriting the logic of their cells, it is possible to make the token propagation logic take advantage of all the 20 ns period of CKREADPIX, doubling the timing margins of this critical part of the chip. Its principle of operation is the following:

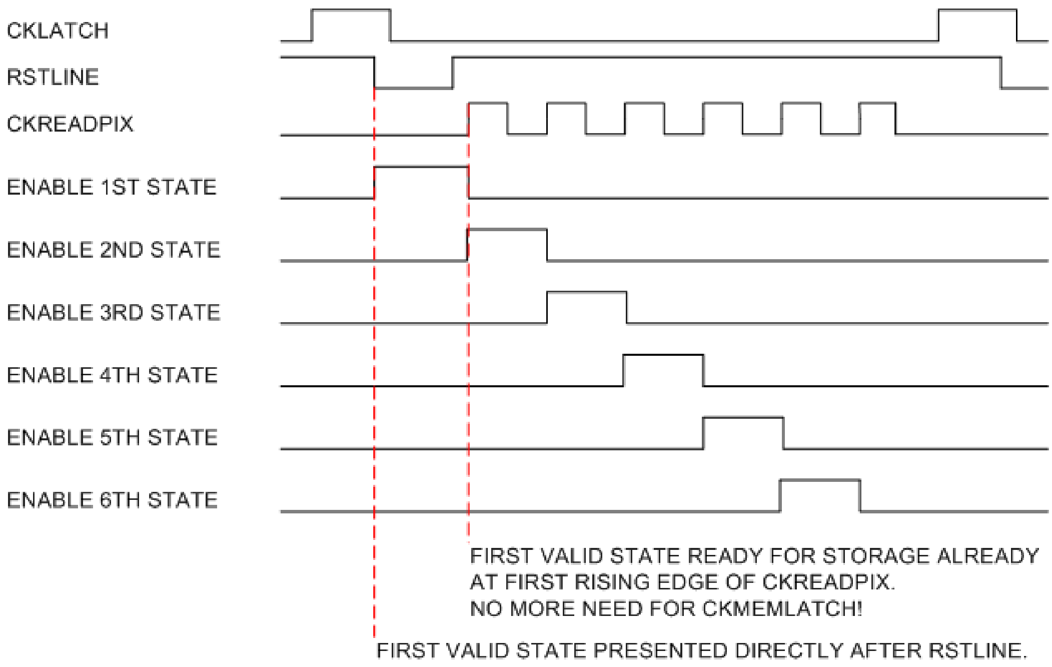

CKLATCH
RSTLINE
CKREADPIX
ENABLE 1ST STATE
ENABLE 2ND STATE
ENABLE 3RD STATE
ENABLE 4TH STATE
ENABLE 5TH STATE
ENABLE 6TH STATE
FIRST VALID STATE READY FOR STORAGE ALREADY AT FIRST RISING EDGE OF CKREADPIX. NO MORE NEED FOR CKMEMLATCH!
FIRST VALID STATE PRESENTED DIRECTLY AFTER RSTLINE.

Figure 3.32: Timing diagram of improved NAND/NOR cells in sparse scan chain.

For each of the pixel cells, there is only one flip flop, at the start of each line reset to zero with RSTLINE. Its negative output (logical one) is AND'ed together with the state validation signal to provide a switch control signal that block the token from propagating to the next pixel cell, and the enable output is generated by once again AND'ing the switch signal together with the token signal. The token signal also acts as enable for a multiplexer providing inputs to the flip-flop, and when it arrives, it switches the multiplexer from pointing to a logical zero to pointing to a logical one. When the CKREADPIX clock arrives, the logical one is loaded into the flip-flop, with the result that the switch control signal routes the token to the next pixel cell while the enable signal is deasserted. See Figure 3.31.

The new implementation simplifies the logic, reduces the number of flip-flops per cell from two to one, and reduces the number of clocks from two to one. With the enable signal staying asserting during one whole clock period, it is also possible to make the registers in the next stage sensitive to the same clock, thus also removing the need for the CKMEMLATCH clock. In total three different clocks are therefore replaced with one clock, and the timing margins are made twice as good. Figure 3.32 shows the new timing diagram.

The new cells have been successfully simulated both behaviourally and at gate level, and they are candidates for implementation in future versions of the design.

3.7 Perspectives for STAR and CBM

Following the successfull tests of the MIMOSA26 chip, three designs are planned as a continuation of this architecture. The two first ones are the Ultimate-1 and Ultimate-2 chips, which will equip the STAR vertex detector. The third one is a double-sided readout chip, MIMOSIS-1, which is to be installed in the MVD vertex detector of CBM. All these chips will be realised in the AMSC35 process.

For ILC, it has been proposed to rewrite the memory management to store 40 frames during the 1 ms long bunch train and perform readout in the 199 ms long dead time. For the outer layers of ILC, it is also intended to redesign the MIMOSA26/Ultimate architecture to interface column-level ADC circuits instead of column-level discriminators. These issues are further discussed in Appendix E.

3.7.1 STAR: Twice as large matrix and higher data transmission rate (Ultimate-1, Ultimate-2)

Ultimate-1 will be a remanufacture of MIMOSA26, with the size of the pixel matrix increased to 1024x1088 and the memories increased from 2x600 words to 2x2048 words (memory increased to store all the hits of STAR 500).

With Ultimate-1, the frequency of the system clock increases from 100 MHz to 160 MHz or 200 MHz. The system clock is divided by two to process the lines in the matrix at 200 ns or 160 ns, while the serial data transmission runs at the full system clock frequency. With twice as high matrix, and twice as many words serial transmitted per line, the amount of data transferred per frame therefore becomes four times as large as for MIMOSA26.

If necessary, there will also be a second version, the Ultimate-2. A PLL for multiplying up a slower external clock to 160 MHz is to be implemented in this chip. For the digital part of Ultimate-2, it is planned to replace the current NRZ encoded serial transmission with an 8b10b encoded serial transmitter.

Both the PLL and the 8b10b encoding circuitry has already been successfully implemented as optional features of MIMOSA26.

3.7.2 CBM: Twice as high frame readout rate (MIMOSIS-1)

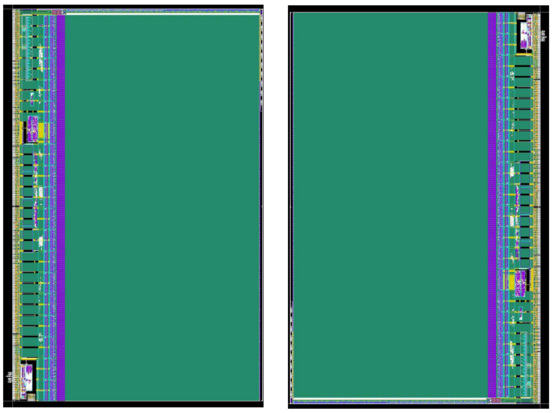


Figure 3.33: MIMOSIS-1. Double-sided readout chip. Two matrices, each of dimensions 256x1152, are read out from each their side [63].

The double-sided readout chip is an attempt to meet the very tight frame readout requirements of the CBM MVD by reducing the number of rows from 576 to 256, and then arrange two such structures back to back so the two chips in total comprise a matrix of 512 rows [63]. The concept is illustrated in Figure 3.33. With this approach, the frame readout rate is expected to come down to 40 µs. If this chip is manufactured in a faster 0.18 um technology, the frame readout time is expected to be reduced even further, down to 25 µs - which in fact meets the frame readout rate requirements for ILC, but still is at least twice as much as the required frame readout time for CBM.

Chapter 4

3D ASICs for pixel detectors

The technique of 3D integration is a combination of three different techniques mature enough to be combined for forming several wafers into one monolithic circuit:

- Stacking. Silicon wafers of same or different processes are stacked on top of each others and bonded mechanically and electrically.
- Through Silicon Vias (TSV). Silicon vias are etched through the entire silicon to provide electrical connections between more than two wafers.
- Thinning. The required diameter of the TSVs is proportional to their length, and the wafers are thinned to obtain vias that consume as little silicon surface as possible.

The current tendency of CMOS scaling is that interconnect delay becomes dominating with respect to transistor delay. At 0.18 µm, the interconnect delay of aluminium wires surrounded by SiO_2 is more than twice as large as the gate delay. At 0.13 µm, the wire delay exceeds the gate delay even for copper wires surrounded by low-K dielectrics. This tendency is critical, as it means that going to smaller dimensions no longer guarantees any improvement in performance.

3D integration, where modules can be stacked on top of each others instead of being spread out on the same die, minimizes the interconnect lengths and may therefore become crucial for continued scaling performance of sub-micron CMOS technology [64].

Applications of particular interest to industry includes small form factor and high capacity memories for portable devices like mobile phones. Memories with low activity - and hence, low power dissipation - are ideal 3D integrated applications [65] [66]. Another example is stacking of memory and processor on top of each other in order to bridge the processor-memory performance gap [67] [68].

For high energy physics, 3D integration provides opportunities of designing detectors where the pixel cell is composed as a 3D stack with separate detector tier[1], analog signal conditioning and processing tier, and one or more digital tiers for storage and readout. Each function can be manufactured in the most optimal process, and with the increasing amount of logic per pixel, fast readout architectures with data sparsification can be developed.

4.1 Motivation for 3D integration in high energy physics

4.1.1 Requirements of future pixel detectors

Important requirements for pixel detectors in future high energy experiments are (1) high time resolution, (2) high spatial resolution, (3) high radiation tolerance, and (4) low material budget (the specific requirements for CBM were presented in Chapter 1).

Without architectural or technological improvements, it becomes difficult to obtain simultaneous improvement in both spatial resolution and time resolution, since these requirements are conflicting. While higher time resolution requires more logic per pixel to provide an architecture with faster readout, a higher spatial resolution imposes a smaller area to implement the increased amount of logic, and also a larger quantity of pixels to read out during a shorter time interval.

4.1.2 Technological features offered by 3D integration

3D integration is a technique that has the potential of overcoming many of the limitations of traditional pixel architectures like hybrid and MAPS.

Compared to hybrid detectors, which traditionally have had pixel pitches an order of magnitude above those of MAPS (like the 250 µm of FE-I3 compared to the 18.4 µm of MIMOSA26), 3D integration provides the following improvements:

- A larger interconnect density can be obtained, and therefore also a smaller pixel pitch, as the electrical connections between wafers are provided by thin TSVs and contacts of the top metal layers (like for example a bonding pitch below 2.4 µm in Chartered-Tezzaron technology [69]), instead of the traditionally large solder bumps of hybrid architectures. Traditional industrial processes are not able to provide bonding pitches below 100 µm [24].

[1]The word tier is used instead of layer in 3D integration terminology

- While bump bonding is limited to providing a face to face connection between two wafers, 3D integration with its through silicon vias makes it possible to attach even a third or a fourth wafer to the stack. Utilizing several tiers, the pixel pitch can be reduced to the desired level while the amount of logic per pixel is kept constant.

Compared to MAPS, which traditionally has frame readout times orders of magnitude larger than those of hybrid detectors (like the 100 µs of MIMOSA26 compared to the 25 ns of FE-I3), 3D integration gives the following improvements:

- For a detector tier that remains standard MAPS and thus limited to NMOS transistors, one or several pixelized wafers can be stacked on top of it to provide the pixel cell with functionality that requires both types of transistors.

- For the same pixel pitch, the available area per pixel becomes proportional to the number of wafers in the 3D stack. By stacking the logic of the pixel cells into different tiers, it is possible, even for pixels of pitches similar to those of MAPS ($< 20\,\mu\text{m}$), to design intelligent pixels with features similar to the readout architectures of current coarse-grained pixel detectors (like the bumb-bonded FE-I3 readout chip for ATLAS).

3D integrated detectors enable stacking multiple sensor and electronics wafers on top of each others and selecting the optimum technology for each layer. With 3D integration, it is possible for example to have a bottom wafer containing the sensor, then another layer of analog electronics, and then a digital layer containing the data sparsification circuitry etc [70].

By placing analog and digital electronics on each their tiers, one minimizes crosstalk between analog and digital signals, since there is no common substrate, and also several metal layers for shielding the two tiers to each other (with the 6 metal layers in Chartered-0.13, a face-to-face bonding in Chartered-Tezzaron process gives 12 metal layers of separation). As there are currently no good analog models, and hence no good analog circuit solutions, for feature sizes below 100 nm, the separation of analog and digital also gives the possibility of utilising a purely digital process for the digital part, with feature sizes below 60 nm [71].

3D integration also gives the possibility of improving material budget, since 3D detectors with the readout electronics put on top of the pixel matrix do not need overlapping sensors to cover inactive area.

4.1.3 3D integrated architectures for providing fast, intelligent small-pitch pixels

For detectors based on rolling-shutter operated MAPS, high time resolution and high spatial resolution are conflicting requirements, since the frame readout time is proportional to the height of the pixel matrix. However, small-pitch pixel architectures based on 3D integration open the perspectives of improving both features at same time:

- With a pixelized readout structure on top of the detector tier, one can implement data driven architectures where the individual pixels autonomously detect their hits, and with their own logic for immediately reporting the hits to the readout circuitry - for example by using timestamp circuitry and token passing.
- Alternatively, one can split the rolling shutter circuitry of MAPS into as many segments as required for a given time resolution - as the 3D integration technology allows placing the zero suppression circuitry on top of each of the segmented rolling shutters.

The world's first 3D integrated pixel detector, a three-tier demonstrator prototype for ILC with a 64 x 64 pixel array, was designed at Fermilab and manufactured in a 0.18 μm SOI multi-project run at MIT Lincoln Labs [73]. The pixel pitch was 20 μm. The design contains the following features:

- Bottom analog electronics tier with integrator and discriminator.
- Intermediate digital electronics tier with 5-bit time stamping.
- Top digital electronics tier with sparsified readout.

In this first prototype there is no separate sensor included - instead the test injection charges are coupled capacitively to the preamplifiers.

In Figure 4.1, a simplified block diagram of a pixel cell is shown. The chip is operated in an acquisition mode where it acquires hits, and a sparsified delayed readout mode where the timestamps of the hit pixels are read out. When a hit arrives during acquisition mode, the pixel cell stores two samples and a timestamp value, and the hit latch in the sparsification circuitry is also set. During the sparsified delayed readout, the pixel cell will then capture a token that is passing through the pixel row, and time stamp and analog values are passed to the end of the pixel column before the token is released by the pixel cell.

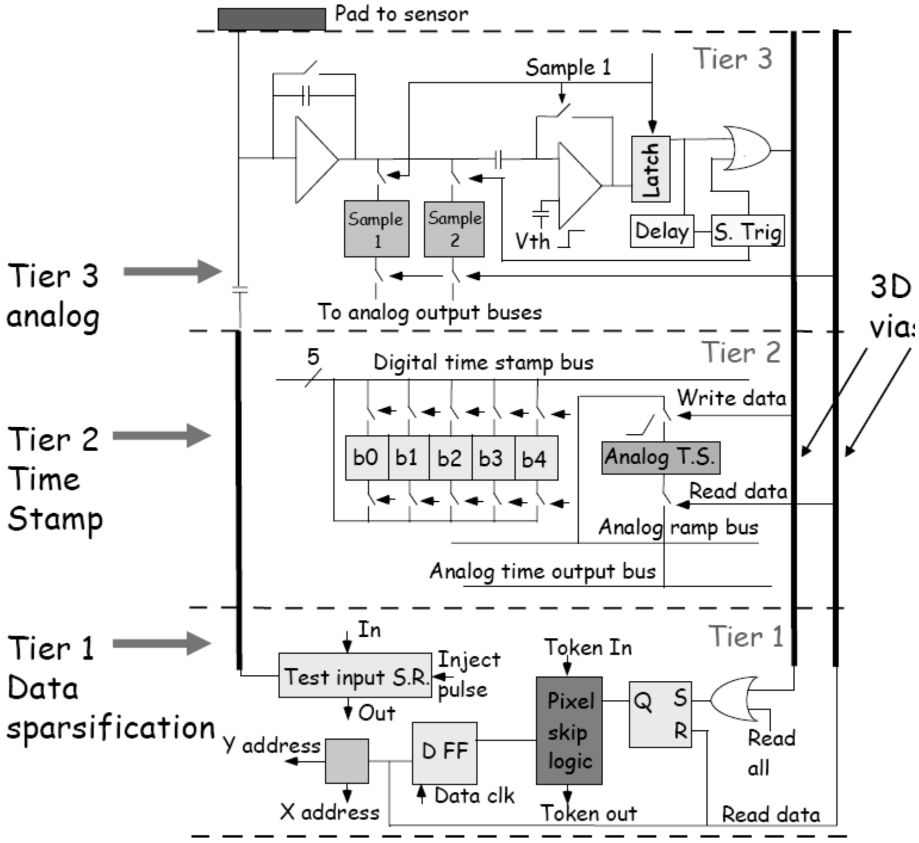


Figure 4.1: Functional block schematic of a pixel cell in the 3-tier Fermilab demonstrator chip [72].

Although the time stamping and sparse readout is performed inside the individual pixel cell, the hit address encoding is still performed by readout electronics located in the perimeter of the pixel array.

As the first demonstrator chip was found to be functionally correct, two later versions have been designed with the purpose of improving yield, increasing the time resolution and to scale with a larger pixel matrix, with the last version being implemented in a 2-tier Tezzaron-Chartered 0.13 µm CMOS process [74].

4.2 Types of 3D integration techniques

Currently, there are different lines of 3D integration, where some technologies have grown more mature than others. The different techniques can be distinguished on their mounting, bonding, and TSV creation. The various techniques will be described in more detail.

Face up mounting or face down mounting

With face up mounting, the wafer to be integrated is first mounted on a handling wafer and thinned from its backside before being mounted on the wafer stack. While with face down, the wafer to be integrated is mounted directly with its front side on the wafer stack, and the thinning of the backside is performed after the wafer is mounted at the stack.

If the bottom wafer of the 3D stack is left unthinned to provide mechanical support for the rest of the 3D stack, it has to be mounted face up to interface with the next wafer. If the second wafer is mounted face down, the bonding between the two wafers is called face to face bonding - otherwise it is called face to back. While face to face represents the simplest way of integrating two wafers, not needing any through silicon vias (TSV), face to back is necessary for the third and following wafers.

Adhesive bonding, oxide bonding, eutectic bonding, thermocompression bonding, or Direct Bond Interconnect

With adhesive bonding or oxide bonding, mechanical bonds with excellent properties for handling and further processing are created. However, no electrical bonds are established, and the electrical connections have to be created afterwards by thinning the top wafer and etching through silicon vias down to the metal layers of the bottom wafer.

With processes using CuSn eutectic bonding or Cu thermocompression bonding, the electrical and mechanical bonds are created simultaneosly. The mechanical bond strength of such bonding is dependent on the percentage of surface used for bonding, with most applications having a typical fill factor of 75 %. The large amount of metal can for some applications cause large parasitics capacitance [75], and may for HEP applications also represent a material budget problem [76].

An alternative to eutectic bonding or thermocompression bonding is the Direct Bond Interconnect, DBI. With DBI, oxide bonds between devices form an immediate bond, but after these bonds have reached sufficient strength, the devices are heated to form metallic thermocompression bonds. With this process, only a minimum of metal needed for the electrical connections is provided, and an extremely low percentage of the radiation length X_0 can be obtained [75]. DBI can be utilized in both face to face and back to back configurations, and it can replace bump bonding in a face to face configuration [75].

FEOL vias, BEOL vias, first vias, or last vias

Back End Of Line[2] vias and Front End Of Line[3] vias are created as part of the wafer processing. While BEOL vias extend all the way through both the silicon and the metal layers, the FEOL vias should be considered the most convenient ones from a designer point of view, as the metal layers above the through silicon via are not occupied by the through via. FEOL vias can also be made thinner than BEOL vias, as they have shorter length and therefore also shorter diameter for the same aspect ratio.

First vias and last vias are vias that are created as part of the 3D integration process. With first vias, the TSVs are created after the wafer processing, but before mechanically and electrically bonding to the 3D stack. While with last vias, the TSVs are etched through the top wafer and the bonding layer after two wafers have been mechanically bonded together.

The longer the through-vias become, the wider they also have to be to maintain the aspect ratios required by the design rules. For all 3D technology with TSVs it is therefore necessary to thin the wafers before etching vias through the wafer. Alternatively, one first creates a blind via from the front side and then thins away enough silicon from the back side to open the via. For detectors this thinning also gives the advantage of improving material budget. Through vias typically have aspect ratio of 8 to 1, and especially for pixel designs it is important that vias occupy as small area is possible (otherwise they would limit the pixel size).

[2]These are the processing steps performed from the first metal layer up to the passivation layer.

[3]These are the processing steps performed up to the first metal layer.

4.3 3D integration suppliers

Currently, there are numerous providers of 3D integration. An overview of different vendors and the techniques they utilize can be found in [64]. In the following comes a short presentation only of the vendors relevant for the 3D integrated pixel detector projects that are referred to in this chapter:

MIT Lincoln Labs

At MIT Lincoln Labs, SOI wafers are connected face to face on the first layer and face to back on the next layers. The wafers are mechanically bonded together using conventional oxide bonding techniques. For each layer that is processed, the handle SOI wafer on top the top layer is thinned completely away, down to the buried oxide, and the inter layer vias are etched through the oxide to make the electrical connections (via last). These vias are about 1.2 µm in diameter.

Ziptronix

At Ziptronix, the already described DBI technique is used to make the electrical connections. Contrary to MIT Lincoln Labs, the electrical connections are formed during bonding, and no through silicon vias are required as long as the 3D stack only consists of two wafers. If TSVs are needed, it is recommended to obtain the wafers from the Chartered-0.13 process.

Normally, more than 95 percent of the bond between two wafers will be silicon and not metal when using the DBI bonding technique of Ziptronix, and bonding pitches less than 1 µm are achievable. A wafer bonded to a 3D stack using DBI can be thinned down to 10 µm [77].

Fraunhofer IZM

Fraunhofer IZM-Munich provides bonding between wafers using a technique called Solid-Liquid-Interdiffusion, SLID. SLID is a technique where metals with high melting point, i.e. Cu, and low melting point, i.e. Sn, are brough together under high pressure and a temperature of around 300 °C (before soldering, the tin has been electroplated onto the copper contacts that are to be brought together). As the liquid Sn diffuses into the still solid Cu, a Copper-Tin intermetallic compound, Cu_3Sn, is created. This intermetallic is stable, and although formed at a relatively low temperature, it has a melting point of 600 °C.

The via formation of Fraunhofer IZM-Munich is via first, as the vias are first etched 12 µm into the silicon before the silicon is thinned at a handle wafer and then bonded to the 3D stack [78].

Tezzaron

With Tezzaron, the wafers are connected face to face on the first layer and face to back on the next layers. The bonding is performed with Cu-Cu thermocompression, of which one of the advantages is its low mass. As the wafers are mounted face down and with the through silicon vias created before bonding, the rest of the silicon must therefore be thinned away after bonding to expose the TSV on the next layer.

Most wafers used by Tezzaron are manufactured by the Chartered-0.13 process. The Chartered process provides through silicon vias, branded as ”supercontacts”, formed after FEOL and before BEOL processing. The supercontacts are typically made of tungsten (as this is the metal typically used at this stage of processing).

The through-silicon vias have a length of around 6 µm (1 µm through the silicon oxide of the first metal layer and 5 µm into the silicon substrate), and since they are made of tungsten, they also tolerate a higher aspect ratio than similar vias made of copper [64] [79]. Similar to the TSVs of MIT LL, the supercontacts are normally around 1.2 µm in diameter. The inter wafer bonds are nominally on a pitch of 4 µm.

Tezzaron also provides its own post-processing TSVs (first vias), branded as ”‘supervias”’. The supervias are made of copper. These vias must span the entire metal stack before extending further 5 µm into the silicon, and to maintain a conservative aspect ratio, they have a diameter of 4 µm.

Both the FEOL and BEOL techniques offered by Tezzaron have their advantages and disadvantages. Although the supervias have a much larger diameter, which may constrain the density of the interconnects, the copper vias have lower resistivity than the tungsten vias - and larger heat conductivity. For signals, the higher resistivity of tungsten is of no significance, but the resistance might become an issue for the power distribution. Figure 4.2 shows a Tezzaron two-tier stack (with BEOL TSV) being ready to be bounded with a third tier. Three-tier stacks with wafers thinned down to 5.5 µm have been demonstrated [69].

4.4 From hybrid detectors to 3D

Hybrid pixel detectors are bonded to their sensors using bump-bonding. With traditional bump diameters of 100 µm or more, the size of the bumps alone could be considered a limitation on the minimum pixel size of such detectors. Bump-bonding has also been limited to face-to-face connections between two wafers. For these reasons, steps have been taken to move current hybrid architectures into 3D.

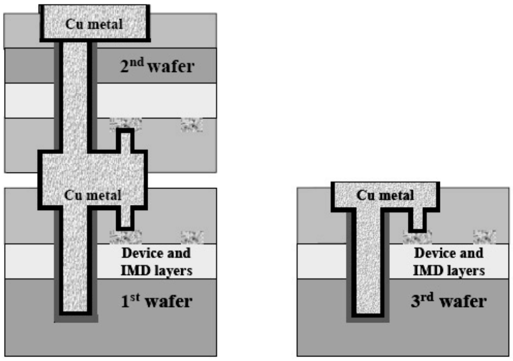


Figure 4.2: Illustration of a Tezzaron two-tier stack which is about to be bounded with a third tier [79].

And the FE-I3 readout ASIC, that is bump-bonded to the ATLAS planar detectors, is step by step being adopted to 3D integration.

However, also in the field of bump bonding, there has been an ongoing development, and currently bumps with diameters of 10 µm or less are available. Still, the mechanical strength of such bonding is not at the same level of the metal-fusion processes used in 3D integration, and this makes thinning and stacking of several wafers difficult [71]. But a combination of micro bump bonding and adhesive injection in the voids between the bumps has recently been demonstrated, with the adhesive providing sufficient mechanical strength to build wafer stacks with up to 40 wafers interconnected through TSV's. Micro bump bonding combined with adhesives and TSV's is therefore also becoming a 3D integration technology [80].

While the original FE-I3 readout chip was designed in a 0.25 µm process [43], a redesign called FE-I4 was performed in 0.13 µm process to reduce the pixel size and to increase the overall active area of the chip [81]. At the same time, the already existing FE-I3 architecture is used to explore the utilization of SLID bonding and TSV formation offered by Fraunhofer IZM-Munchen. As a first step, the bump-bonding is replaced by SLID bonding, while as a second step, the readout chip is thinned down to 50 µm and through silicon vias are created to provide the readout ASIC and the underlying sensor with signals that previously have been provided in an inactive cantilever area of the readout chip [82].

Then finally, there is another 2-tier chip, FE-C4, submitted to Chartered/Tezzaron, where the analog and digital part of the pixel cell electronics is put on each their tier. Partitioning analog and digital reduces both noise and pixel size. The 2-tier chip will be bonded to the detector using the SLID technology developed at

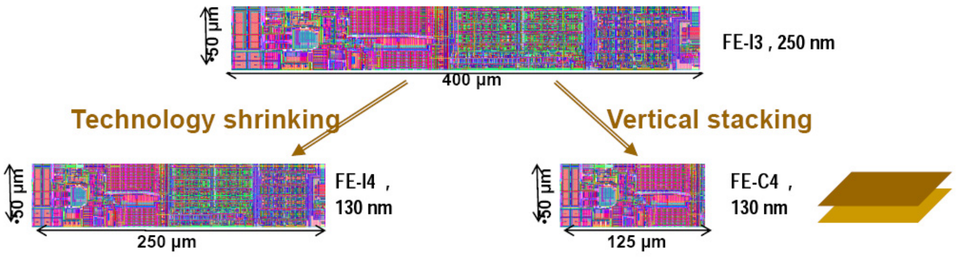


Figure 4.3: Development flow for the ATLAS 3D pixels [83].

Fraunhofer IZM-Munich [83]. See also Figure 4.3.

While the original pixel size of FE-I3 was reduced from 400 µm x 250 µm to 250 µm x 130 µm by going to the smaller technology used in FE-I4, the pixel size was reduced even further, down to 125 µm x 50 µm, when going to the 2-tier 3D integrated FE-C4 readout chip. However, these dimensions are still considerably larger than the small dimensions available in MAPS technology.

4.5 From MAPS to 3D - Internal IPHC prototypes

In late 2008, a consortium composed of a large number of institutions was formed to take part in the first 3D multi project run targeted at applications for high energy physics. The consortium purchased an order with Tezzaron to manufacture vertically integrated circuits using the Chartered 130 nm low power process, CHRT013LP. The circuits have two tiers fabricated with a single set of masks, with the two tiers being bonded together using Cu-Cu thermocompression [84].

The multi project run contained more than 15 circuit designs and dedicated test circuits that were submitted in ten major sub reticules. Two designs were for ATLAS pixel upgrades, another design explores a new trigger concept for the CMS upgrade, while other designs explore pixel designs for the ILC, and various MAPS designs. As seen from Figure 4.4, the integration is done on a wafer to wafer level, with one wafer run being used to manufacture the chips needed for integration.

The IPHC contribution to the multi-project run consisted of three designs, one 2-tier design and two other designs that after successful Tezzaron/Chartered integration will attach to a third wafer, serving as detector tier, and thus make up a 3-tier design. The three-tier designs are thinned to arrive at a 3D stack of 50 µm height [86].

The 2-Tier chip uses the deep n-well Chartered-0.13 process for all the layers

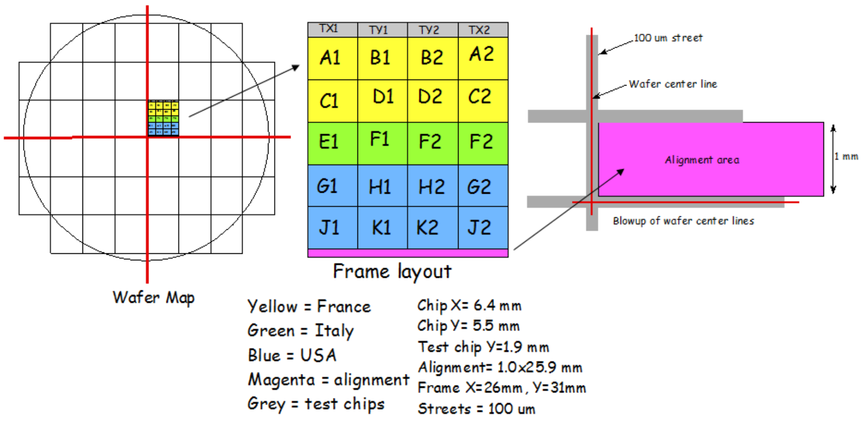


Figure 4.4: Organisation of the first HEP Multiproject Wafer [85].

while the 3-tier designs use Chartered for the two top tiers and XFAB 0.6 high-resitivity epitaxial process (XC06) for a the sensor tier. As the 2-tier chip is face-to-face bonded, without any through silicon vias, only the 3-tier chips are 3D integrated circuits in the true sense of this technology.

The idea is to make the different tiers interchangeable, so that the tiers providing the best results can be combined in future chips. With such an approach, different kind of readout tiers may be assembled together with the same type of detector tier. For example, a low noise photon counting application on three tiers, or a low material budget particle trajectory detector on two tiers.

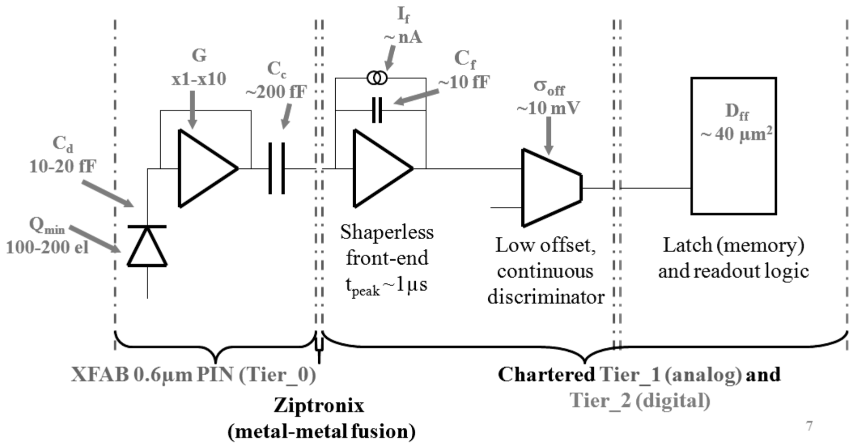
For the digital circuitry, the standard cell library of Artisan Components, which originally was developed for the ARM microprocessor family, are utilised.

With the 3-tier designs, a fully depleted epitaxial substrate with first stage amplifier on the same wafer, is capacitively coupled to the 3D readout electronics on top. While the detector tier is manufactured in the XFAB-0.6 high resistivity process, the two electronics tiers are manufactured in Chartered-0.13 technology. The size of the pixel matrix is 245 x 245 pixels, and the pixel pitch is 20 µm, which gives a total active area of 5 mm x 5 mm.

While the two Chartered wafers are bonded together by Tezzaron, the bonding between the XFAB wafer and the Chartered wafers is performed by Ziptronix.

4.5.1 3-tier architecture: Strip-like

The pixel cell of the XFAB-0.6 detector tier contains a self-biased charge collection diode connected to a source follower. The output of the source follower is then



I_f ~ nA
G x1-x10
C_c ~200 fF
C_f ~10 fF
σ_{off} ~10 mV
D_{ff} ~ 40 µm²
C_d 10-20 fF
Q_{min} 100-200 el
Shaperless front-end t_{peak}~1µs
Low offset, continuous discriminator
Latch (memory) and readout logic
XFAB 0.6µm PIN (Tier_0)
Ziptronix (metal-metal fusion)
Chartered Tier_1 (analog) and Tier_2 (digital)
7

Figure 4.5: Functional block schematic of a three tier design with shaperless frontend [71].

capacitively coupled to the input of a shaperless charge amplification stage (single stage, high gain, folded cascode based amplifier, with constant current feedback) which is located on the second tier. The output of the shaperless frontend is in the next turn connected to a discriminator whose ouput finally connects to the digital readout electronics on a third tier. See Figure 4.5.

For the third tier, a pixel architecture called the STriPSeT (Self Triggering Pixel Strip-like Tracker) has been implemented. With this architecture, a structure similar to that of a strip detector has been implemented. For every column, and for every row, an active low signal is being propagated from the left to the right, and from the top to the bottom, of the chip. For each pixel, the internal hit signal is ANDed together with the incoming hit signal and propagated to the next pixel cell.

In the periphery of the matrix, a fast trigger generator and two shift registers for row and column address identification are implemented. The content of the two shift registers are shifted out of the chip immediately after a trigger signal has been generated. And to mask out noisy pixel, there is also a long shift register chain (JTAG) going through all the pixels [87].

During the time of readout, while the content of the two shift registers is transferred out of the chip, there will be a dead time, whose length is dependent on the operating frequency (for a readout clock of 160 MHz, the array readout time should be approximately 2 µs). Due to the strip-like nature of the architecture, there may also be ambiguities with the localisation in the case of several particles hitting at the same time. However, for experiments with low hit density, like for example a

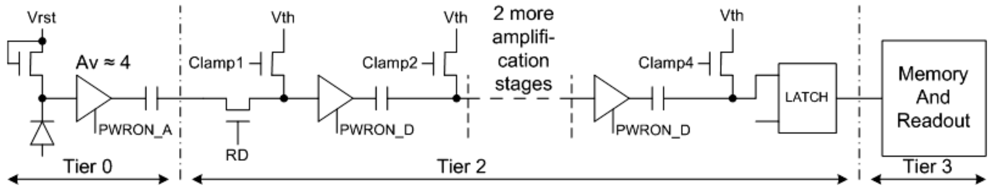


Figure 4.6: Functional block schematic of the 3D integrated rolling shutter architecture. Each pixel has its separate discriminator, implemented at the second tier using a high-gain amplifier composed of four cascaded low-gain amplifiers.

telescope, such an architecture is sufficient for its purpose.

4.5.2 3-tier architecture: Rolling shutter with discriminator in every pixel

One of the major benefits of rolling shutter operation is the low power consumption, as only one row of pixels switches on at each their time. However, a major performance limitation of the current rolling shutter designs is the sharing of the same discriminator between all the pixels in a column.

By embedding a discriminator into every pixel, the line processing speed can be greatly improved compared to the shared column discriminator that is implemented in previous MAPS-based rolling shutter chips. A 3D integrated rolling shutter design, where each pixel contains its own discriminator, was therefore designed by Yavuz Degerli at IRFU [88]. A block schematic of the design is showed in Figure 4.6.

Correlated double sampling is still performed, but the speed is greatly improved since it it is no longer necessary to route the signals far away from the pixels to reach a shared discriminator. In this new design, an in-pixel high gain amplifier at the second tier is using four cascaded low gain amplifiers with offset compensation to obtain sufficient amplification ($A_{total} = A_v^4 \approx 256$) [88].

For 50 ns line processing, it is expected that the power consumption will become 100 µW/pixel. This estimation can be compared with 500 µW/pixel for the 200 ns processing in MIMOSA26 [71]. The design was implemented with a 20 µm pixel pitch and a 32x256 pixel array. To read out the data contained in the second-tier latches, there is a third tier with memory and readout logic.

It should be mentioned that two versions of the rolling shutter chip have been made, one two-tier and one three-tier. For the two-tier chip, one uses standard MAPS where the discriminator circuitry is embedded into the sensor pixel cells, and for the purpose of using the same discriminator in both the 2-tier and the 3-tier versions, the discriminator and latch circuitry has been implemented as NMOS only.

4.5.3 2-tier architecture: Timestamping and delayed readout

Because a deep n-well process is used for the first tier, it is possible to use both p- and n-type transistors for the preamplifier. Instead of the simple low gain amplifiers used in previous MIMOSA chips, it is therefore in principle possible to implement an architecture with a full charge sensitive amplifier with shaper configuration [4].

Due to the small size of the pixel cells, there was no room for a full shaper. Instead of implementing a shaper, the time constant for the charge sensitive amplifier was therefore set low enough to make the amplifier output become a quickly decaying pulse that could be connected to a discriminator.

Preliminary requirements and constraints

The design constraints of the 2-tier IPHC 3D chip are the requirements for the ILC innermost layer. With ILC, the colliding bunches of particles organize into bunch trains where a series of bunches follow consecutively, before a quiet period between the bunch trains. During this quiet period, the inactive detectors have time to cool and the data acquisition system collecting detector data has all the dead time available to receive and process data.

The length of the ILC bunch train is one millisecond, with the bunch trains arriving at a period of 200 ms, each containing 2820 bunches. Due to the periodic shape of the bunch train, an appropriate solution for a pixel detector is to provide each pixel with separate discriminator and time stamp logic, accumulate all the hits during the 1 ms acquisition phase, and perform delayed readout in the 199 ms between the bunch trains. If the hits are binary discriminated and stored with time stamp in the digital tier, the analog tier can be completely powered off during the 199 ns dead time, with the purpose of reducing power consumption.

The concept of operating the detector in a timestamped acquisition mode during the bunch train, and a delayed readout mode between the bunch trains, has already been proposed and implemented in DNW-MAPS with the SDR0 prototype (described in Chapter 2) and in 3D integrated technology with the Fermilab demonstrator chip for ILC (previously described in this chapter). As we will see later in this chapter, the IPHC prototype simplifies the readout mode by replacing the sparsified token-based readout of the two other chips with a scan-chain readout of all the timestamp flip-flops. This implementation simplifies the logic and makes it possible to arrive at the small pixel area of 12 μm x 24 μm, compared to the 25 μm x 25 μm

[4]Although being available, the process step of implanting a deep n-well was omitted in this first 3D wafer run, and the implemented sensor is therefore expected to have a reduced detection efficiency. However, as the purpose of the first prototype was to test functionality, and not to optimize performance, a reduced detection efficiency was considered acceptable.

and 20 μm x 20 μm areas of SDR0 prototype and the Fermilab demonstrator chip. It also enhances the hit detection by adding a second hit flag to the timestamp flip flops.

Architectural concept and prototyping strategy

A 3-tier application, containing a detector tier, an analog electronics tier, and a digital electronics tier, is targeted as the final application for the ILC innermost layers. To meet the spatial and time resolution requirements, the pixel pitch is set to 12 μm[5] and the number of timestamp bits for the 1 ms bunch train is set to 7[6].

However, to make the first prototype fit into a two-tier Tezzaron/Chartered wafer run, the analog electronics splits into two parts that are implemented in each their half of a 12 μm x 24 μm element. The first part contains the charge collecting diode and a common source preamplifier with diode load, while the second signal processing part contains a shaper and a discriminator. The idea is that these two parts can be implemented in each their tier in the final design.

Using two pixel cells for the analog part also gives twice as much space for implementing the digital part of the pixel. However, even with twice as much space, the layout results showed that there was not enough space for more than 5 timestamp flip-flops, and the timestamp circuitry at the digital tier was therefore implemented with a resolution of five bits, corresponding to $\frac{1000\,\mu\text{s}}{32} = 31.25\,\mu\text{s}$.

As the first prototype is primarily designed for validating the functionality of the architecture, and not for optimizing its performance, the compromises in pixel pitch and timestamp bits are considered acceptable. The perspective for the next version of the architecture is then to use a smaller feature size process for the digital part to arrive at the sufficient number of timestamp bits in a pixel cell of 12 μm pitch. Figure 4.7 illustrates the prototyping strategy.

Modes of operation

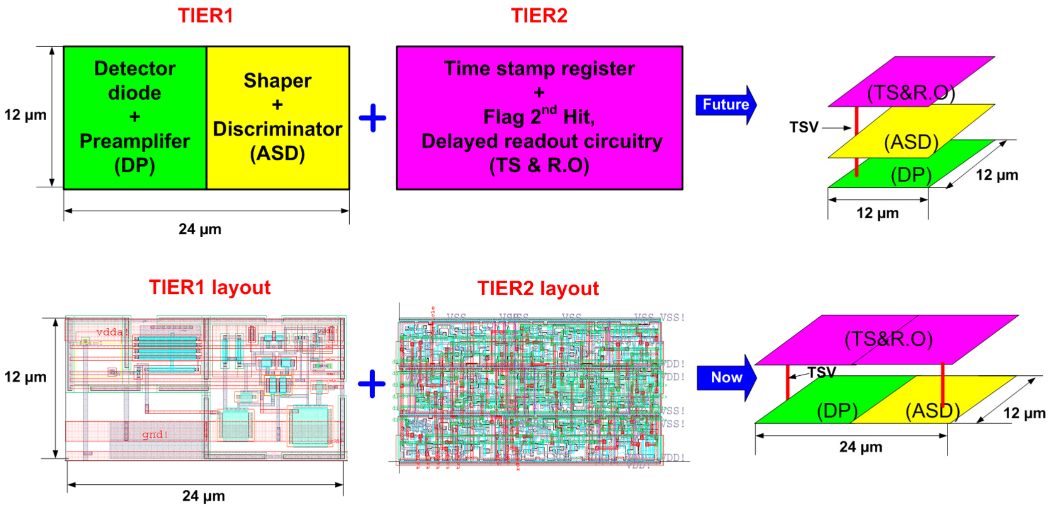
Figure 4.8 shows the external interfaces of the two-tier design, together with the waveform of a normal readout sequence.

As indicated in the figure, the pixel cell may be configured in two different modes:

- Acquisition mode. The particle signal coming from the discriminator of tier 1 triggers storage of the per-column distributed timestamp value. If the particle

[5]For a spatial resolution of 2-3 μm, but also for keeping the amount of second hits at an acceptable level and third hits at a negligible level, as almost all pixel will be hit at least once during the 1 ms bunch train [89].

[6]For a time resolution considerably better than the minimally required 25 us during the 1 ms bunch train



TIER1
TIER2
12 µm
Detector diode + Preamplifer (DP)
Shaper + Discriminator (ASD)
Time stamp register + Flag 2nd Hit, Delayed readout circuitry (TS & R.O)
24 µm
Future
(TS&R.O)
TSV
(ASD)
(DP)
12 µm
12 µm
TIER1 layout
TIER2 layout
12 µm
24 µm
Now
(TS&R.O)
TSV
(DP)
(ASD)
12 µm
24 µm

Figure 4.7: Prototyping strategy for ILC 3D chip.

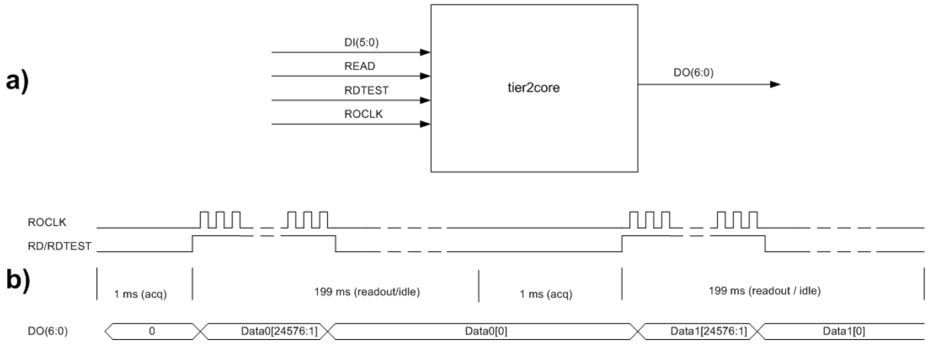


Figure 4.8: a) External interfaces of digital tier. b) Readout sequence of digital tier.

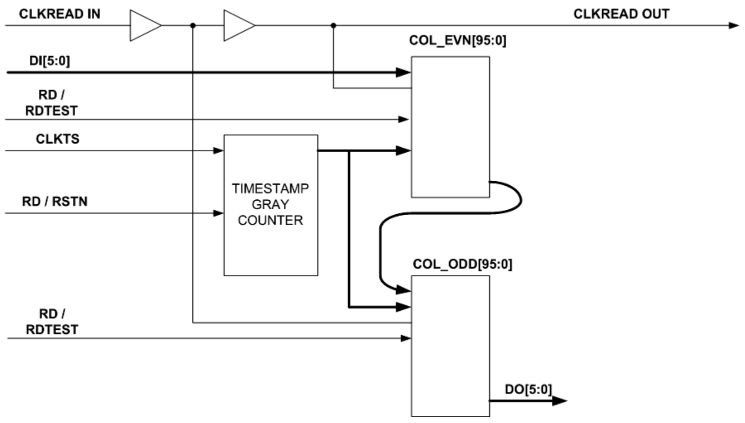


Figure 4.9: Functional block schematic of column pair with gray counter.

signal fires twice, an overflow bit is also set to indicate this second hit without changing the already obtained time information from the first hit.

- Readout mode. The five flipflops in the pixel cells are chained together into serial registers that can be shuffled out of the pixel matrix.

Organisation of the pixel column

For the readout of the 2-tier chip, the timestamp flip-flops are operated as one long shift register. During data acquisition, the flip-flops may be loaded with time stamp information (5 bits, plus 1 overflow bit for detecting a second hit), while during readout, the content of the flip-flops is shifted out of the chip.

Each pixel row has its own Gray counter whose value is distributed to all the pixels in a row. The purpose of using a gray counter is to make sure that only one bit is changing at the same time, since there is no synchronisation between the hit signal coming from the discriminator and the distributed counter. When a hit signal is fired from the analog tier, the gray counted timestamp is stored in the pixel cell. Figure 4.9 shows a functional block diagram of a column pair and its respective gray counter.

Due to area limitations, it is not possible to distribute a balanced clock tree out to all the pixels in the matrix. Instead the clock has to traverse through all the columns from the right to the left, while the data in the shift register goes as a long snake in the other direction, from the left to the right. With this approach, we make sure that data arriving from a previous column is always safely sampled by

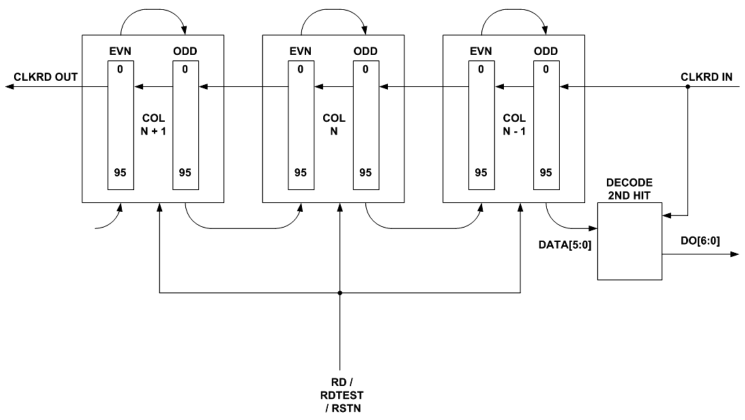


Figure 4.10: Functional block schematic of the organisation of columns and their clock distribution.

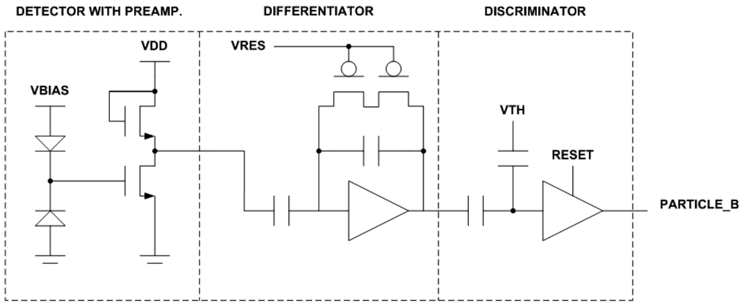


Figure 4.11: Functional block schematic of the analog pixel cell.

the current column before it is once again changed by the previous column.

Figure 4.10 shows a functional block diagram of the organisation of columns and their clock distribution.

Functionality of pixel cells

The analog content of the first-tier pixel cell is shown in more detail in Figure 4.11. At the first stage, charge is collected by the charge collecting diode, and then the generated voltage is preamplified by a common source amplifier with diode load (similar to previous MIMOSA chips). The output of this first preamplification stage is therefore a slowly decaying voltage step.

The next stage in the analog chain is a differentiator with programmable time

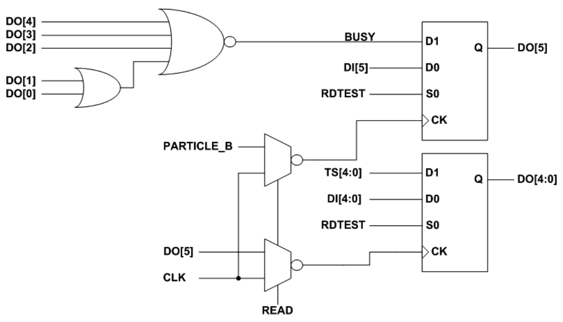


Figure 4.12: Functional block schematic of the digital pixel cell.

constant. The differentiator converts the step function into a pulse that is input to a discriminator.

At the third stage, the discriminator is capacitively coupled to the output of the differentiator. The differentiator input is floating, but can be reset to a stable operating point with a reset pulse. After the reset pulse is deasserted, the threshold value of the discriminator can be adjusted by capacitively injecting an adjustment voltage into the floating input.

The reason of choosing a low-gain NMOS preamplifier with diode load in the first stage, instead of a higher gain amplifier with PMOS load, is to provide a detector and preamplifier part that complies with standard MAPS. In a future configuration, it could for example be an option to use only standard CMOS instead of a process with deep n-well, and thus only utilise NMOS on the first tier while still using both types of transistors on the second analog tier.

The differentiator implemented in the second stage is not optimal for noise. However, the reason for not implementing a full semi-gaussian shaper is the lack of area.

The digital content of the second-tier pixel cell is shown in more detail in Figure 4.12. When the active low particle signal arrives from the analog tier, then the current time stamp value is latched into five flipflops. As seen from the schematic, the particle signal serves as a clock for the upper flip flop, called the second hit flip flop, as long as the circuit is not operated in readout mode.

The timestamp flip flops of the pixel cell are all initialised to zero values, and their outputs are NORed together and fed back to the input of the second hit flip flop. The first time a particle fires, the NORed output of the empty flipflops will be a logical one, and a logical one is therefore latched into the second hit flip flop. It continues through a multiplexer controlled by the read signal and enters the clock inputs of the timestamp flipflop. The current timestamp value is then latched into the timestamp flip flops.

The next time there is a hit, the NORed output of the timestamp flipflops will be zero. The second hit will therefore result in setting the second hit flip flop back to zero value - but without once again triggering the timestamp flip flops, since these are only sensitive to the rising edge of the second hit signal.

When the pixel cell is operated in readout mode, the READ signal and the RDTEST signals are asserted to set the flip flops into scan mode and to route the readout clock into the flip flops.

Test functionality of pixel cells (digital only)

When developing a test strategy for the physical testing of the chip, it is also important to take into consideration that the bonding between the two tiers might be unsuccessfull, and a method to inject the particle signal without being connected to the analog tier is therefore necessary.

Such a feature was implemented by splitting the original READ signal into two separate signals, READ and RDTEST. For normal operation, both signals are asserted at the same time. But while only asserting READ_TEST, and then providing a ROCLK pulse, the current timestamp values are latched into the pixel flip flops. At the next ROCLK pulse, the second hit is also registered.

Data transmission

Although the ILC has 199 ms between the bunches for transferring data, and therefore can afford parallel data transmission at a relaxed data rate, other future experiments like CBM require continuous transmission of data at frame readout rates of a few µs. Such data rates are only realizable using differential serial transmission interfaces with embedded clock. Implementing a differential serial interface with embedded clock does not only enable data transmission rates that exceed the limits of parallell backplane buses - it also requires fewer wires, and thus lower power consumption, better material budget and improved reliability.

For complying with future projects where there is much smaller or no dead time at all between the bunch trains, and with a much higher data rate, we propose to equip the design with an 8b10b serial transmission interface. The 8b10b interface is designed for reuse in later projects, and the interfaces between the 8b10b module and the rest of the 3D chip are therefore equipped with glue logic that enables separate testing of both the pixel matrix and the 8b10b module.

The 8b10b protocol was presented for the first time in 1983 by IBM [90]. Their patent has now expired, and the protocol is free to be used. 8b10b encoding is already implemented in protocols like Gigabit Ethernet and PCI Express.

The main reasons for using serial transmission with embedded clock are:

- The cables going from the detector to the counting room have lengths of several meters. A parallell bus is strongly limited with respect to clock rate, since there are different delays in the different bus wires. A differential serial bus with embedded clock is only limited by the bandwidth of copper, and can deliver data rates into the GHz range.
- We also want the reduce the amount of wires, since these contribute to material budget and power consumption - and since every wire is a possible source of failure, we also improve reliability.
- With a DC-free serial transmission code, the receiver can be connected capacitively, preventing possible ground loops.

Instead of developing an 8b10b encoder from scratch, an existing Verilog implementation of a combinatorial 8b10b encoder module was downloaded from chuckbenz.org. The 8b10b encoder was then embedded together with a state machine controlling both the readout interface of the pixel matrix and the 8b10b encoder. The state machine operates by reading 8-bit data from the pixel matrix, feeding it to the 8b10b encoder and serializing the resulting 10-bit string.

The 8b10b encoder had already been successfully implemented in the AMSC35 opto process, as an optional feature of MIMOSA26. For this implementation, two hardware tests were performed, one at 160 Mbit/second and another one at 256 Mbit/second. For both tests, the bit error rate was below 10^{-12} (personal correspondance with Q. Sun). One should therefore not expect serious problems when the same circuitry is implemented in the even smaller and faster Chartered-0.13 process.

A schematic block diagram of the readout control with embedded 8b10b core is shown in Figure 4.13.

The RO_8b10b architecture contains a data path where data is taken pixel cell by pixel cell and fed into the 8b10b encoder and serialized out of the chip. To test the encoder circuitry independently, there is also a multiplexer in front of the 8b10b encoder that is used to multiplex between the pixel matrix shift registers and a register containing synchronized test data from external pads.

A clock divider (the $\ddot{\text{d}}$iv$\ddot{\text{b}}$lock in Figure 4.13), whose timing waveform is shown in Figure 4.14 with the 8-bit output of the pixel column, is used to provide the necessary clocks for the pixel matrix and the enable signals for the rest of the circuit.

The state machine operates in three different states, the idle state, the preamble state, and the data state. The default state is the idle state, where the 8b10b

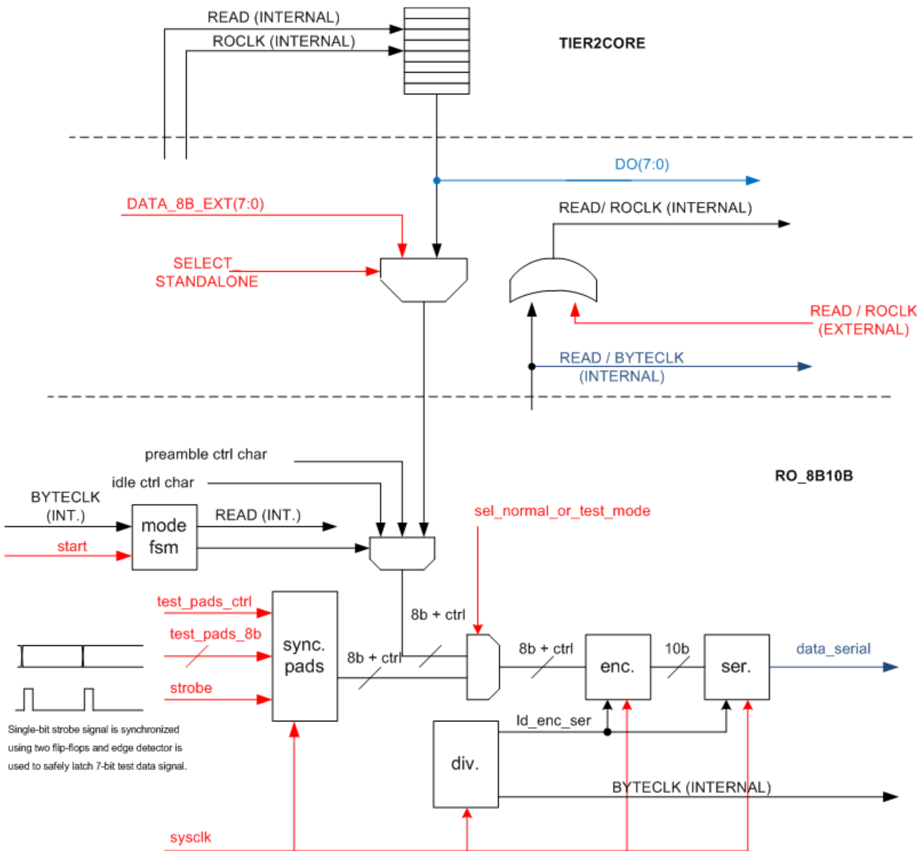


Figure 4.13: Schematic block diagram of the readout control with embedded 8b10b encoder. External input pads are shown in red and external output pads are shown in blue.

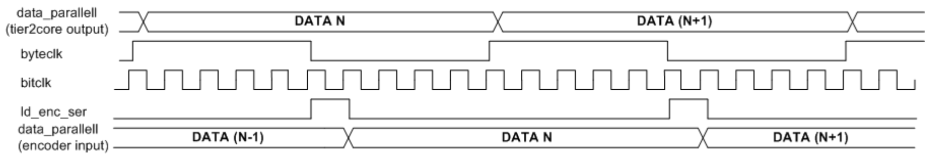


Figure 4.14: Waveform showing how the RO_8b10b and TIER2CORE modules are synchronised with each others.

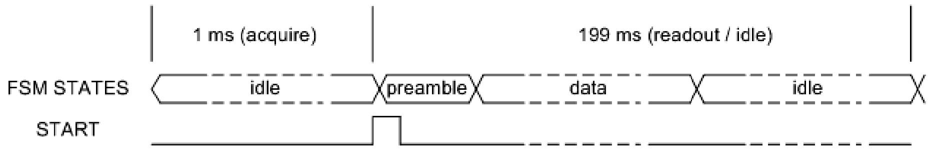


Figure 4.15: Behaviour of the state machine with embedded 8b10b encoder.

encoder is commanded to continuosly give out the comma control character. When the external START signal is asserted, the state machine starts to assert the read signal and provide the readout clock to the TIER2CORE, while at the same time commanding the 8b10b encoder to present another control character as a header. After providing the header, the 8b10b encoder continues by transferring the data provided by the TIER2CORE. See Figure 4.15.

Figure 4.13 shows how the state machine is able to autonomously control the readout, while still also providing full external access to the TIER2CORE through OR gates. Since the state machine is responsible of giving out all clocks and read signals, it is also possible to strongly reduce the readout frequency and use this state machine to control the parallell readout also.

As seen in the figure, the READ and ROCLK signals provided by RO_8b10b will be put out on external pads, in addition to being internally ORed together with the external READ and ROCLK signals. As long as the RO_8b10b is not put into operation by asserting the START signal, it is therefore still possible to fully control the TIER2CORE externally. By ORing the interface signals together, it is also possible to emulate particle hits during the acquisition phase by asserting the external READ_TEST and ROCLK signals. By asserting the SELECT_STANDALONE signal, it is also possible to internally disconnect the TIER2CORE and RO_8b10b modules and stimulate both modules directly from the outside world.

Separate test features for the data transmission

To be able to test the 8b10b module separately, it is possible to disconnect the TIER2CORE data inputs and instead stimulate the data inputs directly from external pads. This mode is enabled by asserting the SELECT_STANDALONE signal.

To provide for a separate test of the 8b10b encoder, it is also possible to bypass the multiplexer controlled by the state machine and directly access the inputs of the 8b10b encoder. This is done by asserting the select_normal_or_test_mode signal. In this mode, the 8b10b encoder can be fully accessed with not only 8 external data pads, but also another external control pad to select if the 8-bit word is to be encoded as data or as a control character.

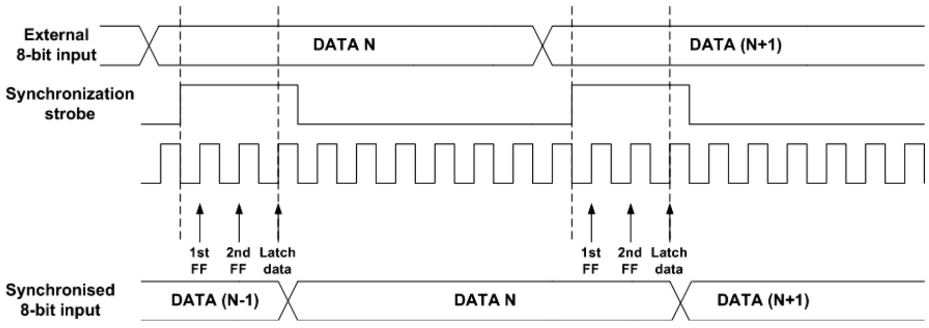


Figure 4.16: Synchronisation sequence of external test pads.

As already seen from Figure 4.13, these external pads are fully synchronized using another external strobe signal. The strobe signal is fed through two flip flops and an edge detector whose output is used to enable loading of the external signal. To ensure that the external input signal is loaded and 8b10b encoded once and only once, the external signal must be provided at the same rate as that of the byteclock, as shown in Figure 4.16.

Timing performance of the data transmission

The fastest IO cells of the Artisan standard library have typical delay of 2 ns (worst 4 ns and best 1.3 ns). To be able to deliver data at the maximum data rate available with these standard cells, an operating frequency of 500 MHz should be targeted.

After performing optimally constrained synthesis, placement, and routing, arriving at a total area of 230 µm x 90 µm, the static timing analysis showed that an operating frequency of 770 MHz was achievable for the typical conditions (typical transistors, 1.5V, 25 °C). With the worst conditions (slow transistors, 1.35V, 125 °C), a frequency of 490 MHz was achievable - but for these conditions, it is the 4 ns delay of IO cells, and not the 8b10b readout block, that becomes the critical part.

The same timing analysis also revealed that the critical path of the readout block traverses from input to output of the combinatorial 8b10b encoder. Further improvement in timing therefore requires an optimization of the 8b10b encoding itself.

Simulation and verification environment

To validate the digital part of the design, including the 8b10b interface, we proposed and performed the following set of simulation tests:

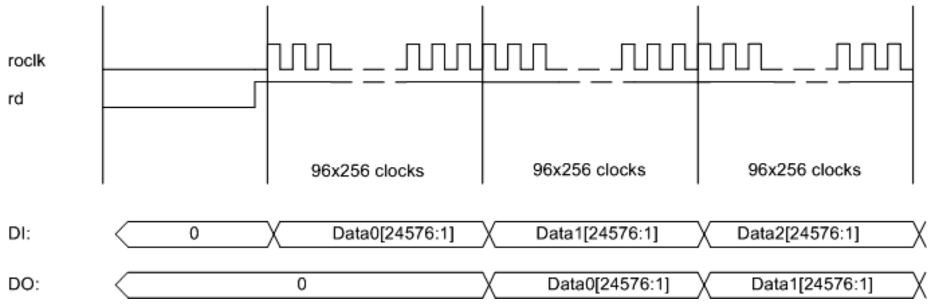


Figure 4.17: Test of the shift register chain. Data comes out after a latency of 24576 clock cycles, which are necessary to cross the shift register.

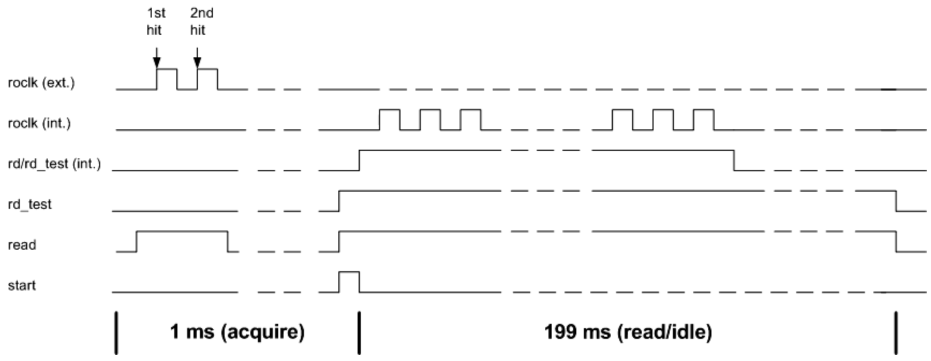


Figure 4.18: Emulation of particle hits using the READ, RDTEST and ROCLK signals.

- Test with the design running in normal mode and particles injected by forcing the particle signal.
- Shifting test, where the chip is operated in readout mode and external data is shifted into the pixel flip flops to evaluate the scan chain. See Figure 4.17.
- Particle emulation test. The read signal is asserted, without asserting read test, to route the readout clock to the flip flops and use this signal to inject a hit instead of the particle signal. See Figure 4.18.

With the first test, a TCL script was used to force the particle signals at appropriate times. The time, the current time stamp value, and the location of the forced hit pixel, was at the same time written to a text file for comparison with another text file generated by a test bench module in the receiver. As long as all the first hit pixels are injected before the second hit pixels start to become injected, it is easy

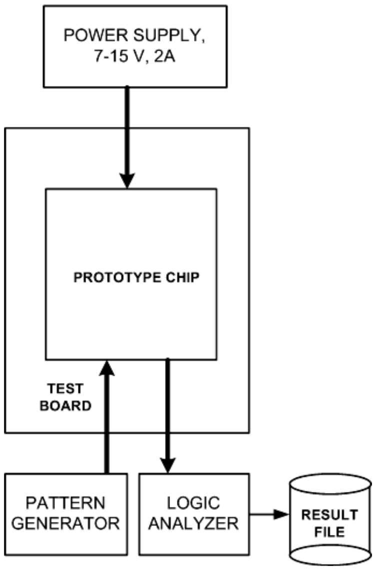


Figure 4.19: Test setup for 3D chip. For this automatized test, one could for example use the Agilent 16822A that is both a pattern generator and a logic analyzer.

to compare the two text files and search for inconsistencies.

For the test, it was chosen to inject one particle at top and bottom of every column, and two particles in the middle of every column. For the shifting test, we may use a counter at the external input to systematically change the values that are loaded into the scan chain of the flipflops.

Physical test

Most of the same tests that we proposed and performed in simulation may easily be performed as actual hardware tests when the chip has returned from manufacture, with the sole exception of the test based on forcing the particle signal.

Figure 4.19 shows the test setup proposed for the 3D chip. A pattern generator is used to provide the necessary clocks and input stimuli and logic analyzer is used to capture the corresponding output of the chip.

There are two ways to synchronise the time stamp counter, either by providing an asynchronous reset right before each bunch train or by providing a synchronous reset. For the synchronous reset, we use the read signal to keep the time stamp counter synchronously reset to zero value until a bunch train is initiated. The two ways of synchronising the time stamp counter are shown in Figure 4.20 and

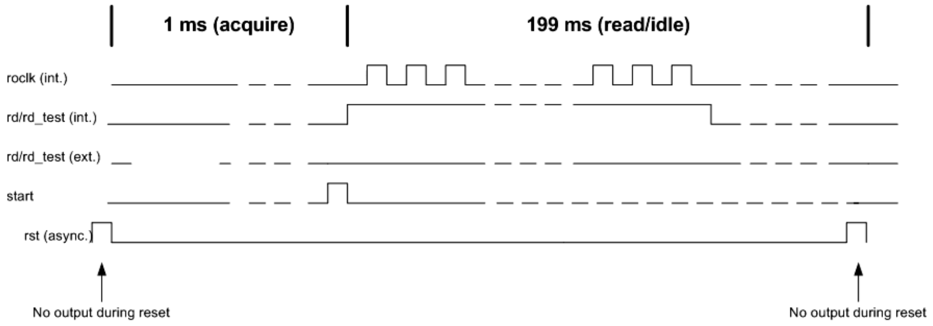


Figure 4.20: Waveform showing asynchronous readout.

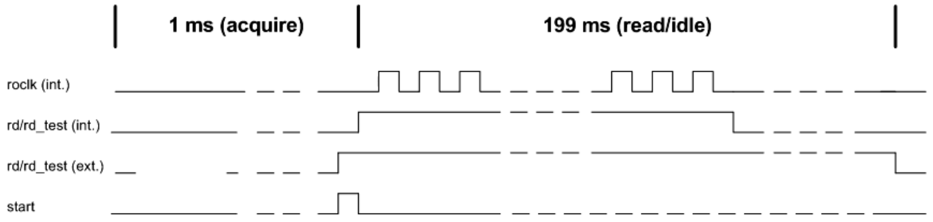


Figure 4.21: Waveform showing synchronous readout.

Figure 4.21.

In Figure 4.20, the timestamp counter is asynchronously reset to zero in front of every 1 ms acquisition period. The disadvantage of using the asynchronous reset in a test setup is that the RO_8b10b module is also reset at the same time, and with a PLL in the receiving end to recover the clock, the PLL has to resynchronise for every acquisition.

In Figure 4.21, there is no asynchronous reset pulse in front of the acquisition period. Instead, the external READ signal, that is ORed together with the READ signal provided by RO_8b10b, is asserted during the entire 199 ms when there is no acquisition. The external READ signal is deasserted right in front of the next 1 ms acquisition period to keep the timestamp counter synchronised with the data acquisition.

To be able to verify if the particle emulation test is successfull, it is necessary for the pattern generator to have its own counter. At a given counter value, a particle hit is emulated. This value has to be compared with the actual timestamp values received from the chip.

For the shifting test, a counter sensitive to the readout clock, which can provide continuously new data inputs to the 3D chip, is required. Alternatively, the values might be read from a text file - and this text file can afterwards be compared with a text file containing the received values.

It is impossible to physically test the scheme of injecting particles into every corner of the columns. The only substitute for this test is to perform a test with real particle source after successfull bonding with the analog tier.

4.6 Conclusion

For MAPS-based applications, 3D integrated pixel architectures have been proposed, with the purpose of improving their time resolution and radiation hardness while maintaining their high spatial resolution ($< 20\,\mu m$ pixel pitch) and low material budget ($50\,\mu m$ substrate depth after thinning).

The first prototypes have been developed, which compared to standard MAPS have improved features both in terms of time resolution ($31.25\,\mu s$ for ILC prototype, compared to $100\,\mu s$ for MIMOSA26) and radiation hardness (where the tolerable amount of non-ionizing radiation expected to increase from 10^{12} neq in standard MAPS to 10^{14} when a high-resistivity epitaxial detector tier is used). The power consumption is also reduced, as a rolling shutter operated 3D MAPS consumes $100\,\mu W$/pixel, compared to $500\,\mu W$/pixel for rolling shutter operated 2D MAPS.

A pixel pitch of $12\,\mu m$ was obtained for the ILC prototype, significantly smaller than the pitches of MIMOSA26. And finally, with the post-processing steps of the Tezzaron and Ziptronix 3D integration, each wafer in the 3D stack can be thinned down to $10\,\mu m$ or less. Even with three or four tiers in the 3D stack, the material budget is still at the same low level as for standard MAPS.

Pixel architectures based on 3D therefore also become the candidate architectures for the most challenging applications like the inner layers of ILC and the first detector station of the CBM MVD.

A readout block with 8b10b interface has been developed for equipping these detectors with fast serial transmitters, capable of transmitting at a data rate of 500 MHz or more.

Chapter 5

Conceptual design for a CBM 3D integrated detector

The reconstruction of open charm is foreseen via their hadronic decays into pions and kaons through the weak interaction. The background of these channels is formed by kaons and pions, produced in large numbers by the primary collision. To distinguish the signal from this combinatorial background, it is necessary to reconstruct the decay vertex of open charm particles. Because of the relatively low lifetime of open charm, a vertex detector with excellent performances is required.

To perform the secondary vertex identification of open charm mesons, a Micro Vertex Detector (MVD), composed by Monolithic Active Pixel Sensors (MAPS), is to be constructed right behind the target. By reconstructing the secondary vertex of the open charm mesons, it is possible to suppress the combinatorial background [91].

The vertex detector has to assure a secondary vertex resolution of approximately 50 µm. With the first station of the pixel detectors located 5 centimeters away from the target, a spatial pixel resolution of 10 µm is required in the ideal case of ultra-thin detectors with no secondary scattering. A more realistic case of 300 µm thick silicon in each detector station imposes a pixel resolution of 5 µm.

With the CBM experiment, running at its full nominal collision rate, 10^9 ions per second pass through a target of 1 % interaction length. This gives approximately 10^7 collisions/second in addition to the delta electrons.

To obtain sufficient statistics for the open charm physics, the experiment should be run with a collision rate of 10^6 collisions/second. It is therefore strongly desired to provide a concept that makes it possible to operate the Multi-Vertex Detector (MVD) at such a collision rate.

Simulation results of first MVD station (obtained in November 2009 from Michel Deveaux), 5 cm from target, show that close to the beam hole, the hit density is dominated by delta-electrons knocked out of the target. For the cases of one million collisions/second, and a readout rate of 500.000 frames/second (frame readout time of 2 µs), small hotspots of 1 mm^2 size may arrive at 7 $particles/mm^2/frame$ (7.2 particles/mm^2 in the tightest hotspot).

As seen from Figure 5.1 a), it has been proposed to arrange the MVD sensors to point away from the beam hole, with only a small part of the sensor covering the hottest areas close to the beam hole. The proposal is based on an arrangement of MIMOSIS-1 chips, where the sensors are located on both sides of the station in an overlapping fashion to cover the inactive peripheral area on both sides of the sensors [92]. Figure 5.1 b) shows a proposal on how to arrange the chips at the same side of the detector station when 3D chips with fully active area are used (the same arrangement can be rotated 90 degrees on the other side of the station to overlap the small inactive areas that should be left between the arranged sensors).

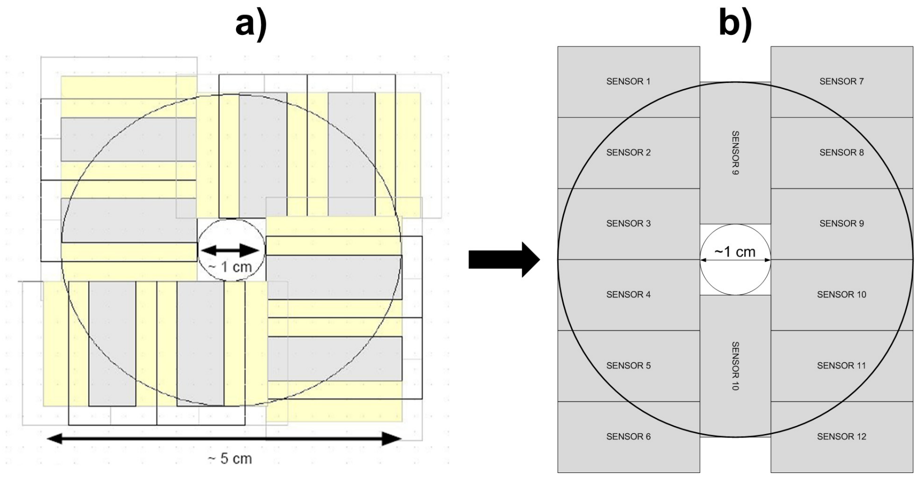


Figure 5.1: a) Preliminary arrangement of sensors around the CBM beamhole. Alternative options may be proposed if found more useful [92]. b) Proposal of arrangement with 3D detectors having close to 100 % active surface.

In Figure 5.2, one sees the simulation results together with what becomes the worst-case arranged sensor in both case a) and case b) of Figure 5.1 (which is not necessarily the final sensor arrangement, but still the starting point for this study). From this arrangement it can be derived that it is necessary with a pixel sensor readout architecture that in average can process and transfer hits from 3.5 particles/mm^2/frame, while also being capable of handling a twice as large hit density (7.2 particles/mm^2/frame) in smaller segments of the chip.

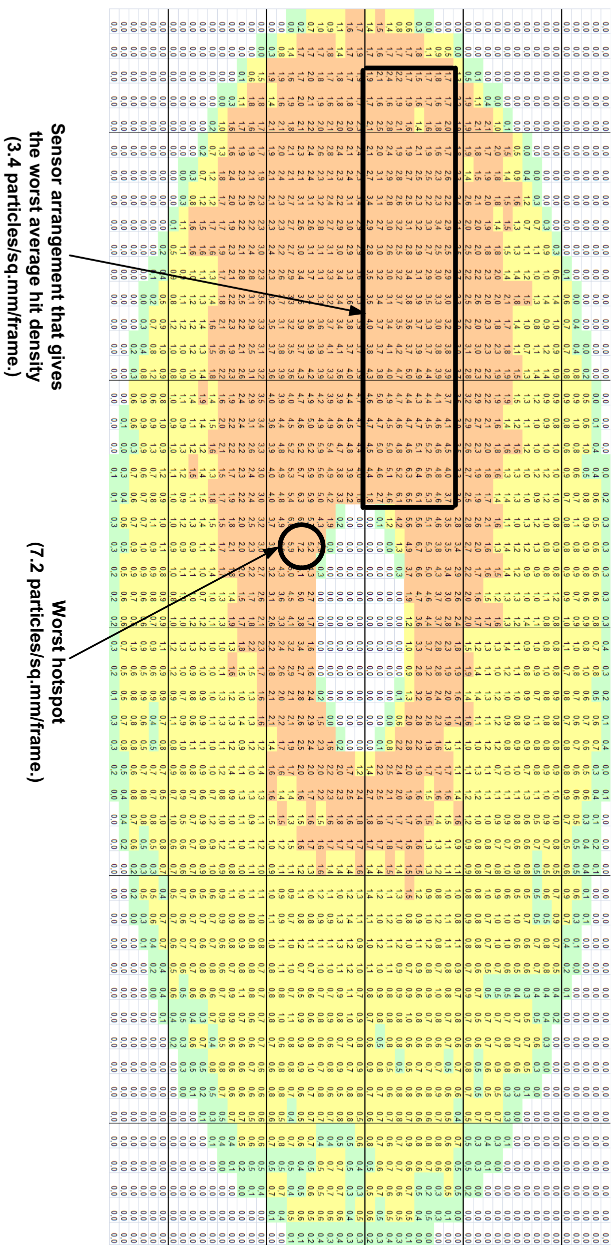


Figure 5.2: Simulation results showing the density of hits per frame with 500,000 frames/second and 10^6 collisions/second. Green: $0.1 - 0.5$ $hits/mm^2/frame$. Yellow: $0.5 - 1.4$ $hits/mm^2/frame$. Orange: ≤ 1.5 $hits/mm^2/frame$.

In the previous chapters, two different approaches for meeting the time resolution have been presented. With MIMOSA26 and the chips that are planned to continue in this direction, the strategy relies on continuous readout with sufficiently fast frame readout rate. While with the first 3D chip for ILC, the strategy was to use timestamping for storing all hit information on-chip during acquisition, and then use the time window between the bunch trains for delayed readout.

For the last approach, based on timestamping, it is also necessary for the experiment to provide a time window between the bunch trains that is sufficiently long to transfer all the data that were obtained during a bunch train. Some experiments, like the CBM, have continuous beam, and delayed readout is therefore not an option for CBM.

With an architecture based on MIMOSA26, implemented in a faster process, it is expected that with a dual-sided readout one may come down to a frame readout rate of 25 µs, thus meeting the requirements for the first generation of CBM MVD detectors with collision rates at 10^5 collisions/s [93]. However, it is still far from meeting the requirements for the later CBM upgrade where collision rate increases to 10^6 collisions/s.

A suitable approach for meeting the CBM upgrade requirements, using 3D integrated technology, is to continue with rolling shutter operation, while at the same time cutting the rows of the pixel matrix into smaller subsegments, where the number of rows in each subsegment corresponds to the desired frame readout time.

5.1 Requirements for a 3D integrated pixel detector for MVD

5.1.1 Spatial and time resolution

For a pure binary resolution (pitch divided by square root of twelve), a maximum pixel pitch of 18 µm is imposed. However, due to charge sharing between pixels, a slightly larger pitch should be allowed if necessary. As the Artisan standard cells have a height of 3.69 µm, it is therefore tempting to set a pixel pitch of 18.45 µm to fully utilize the pixel cell area.

The high level trigger of CBM uses the STS to provide initial tracks, which are refined using the MVD. To separate the tracks, the MVD should not have occupancy above one percent. The required time resolution is therefore determined by the maximum occupancy of the detector.

With a pixel pitch of 18 µm, the number of pixels/mm^2 becomes 3100, and the expected occupancy in the worst-case hotspot of 7.2 $particles/mm^2/frame$ (with

2 µs time resolution) amounts to:

$$Occupancy = \frac{hit\_density \cdot pixel\_multiplicity}{pixel\_density} = \frac{(7.2\, hits/mm^2/frame)\cdot(4\, pixels/hit)}{(3100\, pixels/mm^2)}$$

$$= 0.93\ \%\ per frame.$$

(Worst-case multiplicity of 4 is assumed for the high-resistivity epitaxial MAPS used in CBM MVD, as will be justified later in this chapter).

The occupancy is 0.93 % per frame when the time resolution has come down to 2 µs, and we can therefore afford a frame readout time of 2 µs when the collision rate is 10^6 collisions/sec.

5.1.2 Radiation hardness

Non-ionizing

At the border of the beam hole of the first MVD station, there will be up to $2 \cdot 10^{15} neq/cm^2/year$ when the experiment is run at its nominal collision rate. With collision rate reduced from 10^7 to 10^6 collision/s, the fluence is reduced to $2 \cdot 10^{14} neq/cm^2/year$, and it has already been shown that XFAB MAPS detectors have excellent charge collection efficiency all the way up to $3 \cdot 10^{13} neq/cm^2$ before starting to slightly deteriorate [44]. One assumes that the radiation tolerance of MAPS sensors with high-resistivity epitaxial can be extended to an order of 10^{14} [37]. See Figure 5.3.

With detectors based on depleted epitaxial, it should therefore be possible to extend the radiation tolerance up to the order of 10^{14} neq/cm^2 and thus meet the requirements on non-ionizing radiation.

Ionizing

In the hottest regions of the MVD, the total ionizing dose may reach 340 MRad when the experiments runs at the full collision rate, which means approximately 34 MRad when the collision rate is reduced to 10^6 collisions/sec. It has already been demonstrated that analog and digital electronics designs based on Chartered wafers can withstand ionising radiation doses up to 400 MRad [83].

However, what is still to be investigated is how the non-ionising radiation affects the leakage currents of the detector sensing diode, and thus the input noise of the amplifier at the detector tier. As already discussed, the sensing tier of the CBM 3D device will be manufactured in a technology chosen for its high-resistivity epitaxial layer parametres, like XFAB-0.6 or XFAB-0.35. The feature size of the sensing tier

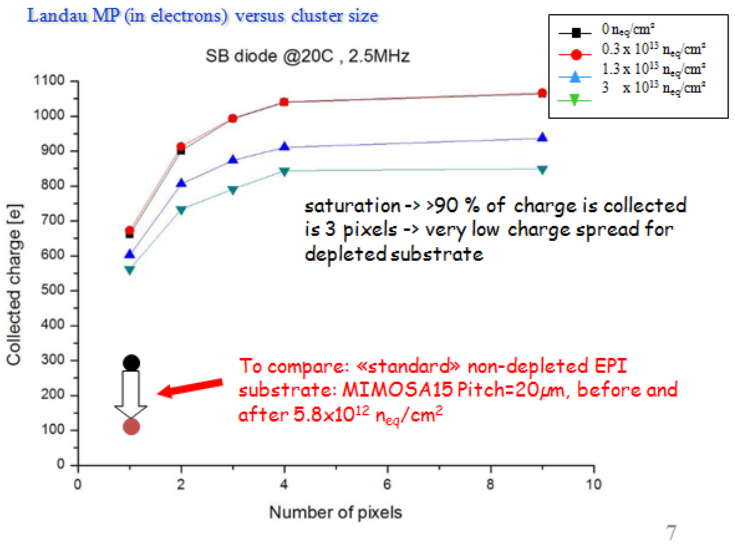


Figure 5.3: Charge collection efficiency versus radiation dose for XFAB high-resistitivity MAPS [44].

will therefore be larger than that of the Chartered-0.13, and the sensing tier is thus likely to be more sensitive to ionising radiation.

5.1.3 Power consumption

The power consumption of the detectors should not exceed 2 W/cm^2. In the worst case, power consumption up to 5 W/cm^2 may be tolerated, but this requires additional cooling and influences the material budget.

Power saving approach: Rolling shutter

With rolling shutter operation of the pixel matrix, only one line in a MAPS sensor is activated at each time, thus saving the power otherwise consumed in the amplification circuitry. To meet a given timing resolution, the number of lines processed by the rolling shutter cannot exceed:

$$Max\ height = \frac{Frame\ readout\ time\ (requirement)}{line\ processing\ time\ (performance)}.$$

The first 3D chip with rolling shutter operation was designed by Yavuz Degerli at IRFU, and has already been presented in the previous chapter. To improve speed and save power, the shared column discriminators of previous architectures are replaced with a single discriminator inside every pixel cell.

The new design is able to operate with 50 ns/line, and with an expected power consumption of 100 μW/pixel. This estimation can be compared with 500 μW/pixel for the 200 ns processing in MIMOSA26 [71]. Due to the new degrees of freedom offered by 3D integration, speed is increased with a factor of four and power is reduced by a factor of 5. It is expected that newer versions of the architecture can be optimized even more, and reach a line processing time of 40 ns (personal correspondance with Yavuz Degerli and [94]).

To improve the time resolution further, the rolling shutter chip splits into segments, and as we will see later in this chapter, we propose to split the matrix into 10 submatrices, with each submatrix composed of 50 rows and 1024 columns. As the next line is powered on while the current line is processed, there will for each segment be 2 x 1024 pixels. With 10 segments, the instantantenous power per cm^2 becomes:

$$\frac{Matrix\ power}{Matrix\ area} = \frac{10 \cdot 2 \cdot 1024\, pixels \cdot 100\, \mu\mathrm{W}/pixel}{18.4\, \mu\mathrm{m} \cdot 18.4\, \mu\mathrm{m} \cdot 1024 \cdot 500} \approx \frac{2\ W}{1.6\ cm^2} = 1.25\ W/cm^2.$$

With a power budget of 2 W/cm^2, there is 0.75 W/cm^2 left to spend for the remaining electronics (digital readout), corresponding to 0.75 $W/cm^2 \cdot 1.6\ cm^2 =$ 1.2 W/chip.

5.1.4 Data reduction

With standard MAPS, the assumed number of fired pixels per particle is five, and thus with 3 states necessary to code a regular cluster. Compared to state encoding, which for such a cluster requires three states, an on-chip clusterisation algorithm is therefore clearly beneficial.

With fully depleted MAPS, much smaller clusters are assumed. The first available test results for MIMOSA25 showed average multiplicity of 4, and an average number of lines per cluster of 2 (data analysis performed by R. De Masi).

The referred data analysis was performed offline with a cut of 80 ADC counts (approximately 120 electrons) to search for seed pixels and a lower cut of 40 ADC counts to search for surrounding pixels. The search for surrounding pixels in a lower threshold range is performed to improve the resolution. However, such search cannot be performed for a detector with embedded zero-suppression, as the discriminators will have a threshold that is common for all pixels.

If the search for clusters and the search for surrounding pixels is done with the same threshold of 120 electrons, it is obvious that the clusters will be of smaller size than the results obtained from the dual-threshold data analysis. When the work with the conceptual design started, one therefore considered the cluster sizes obtained from the MIMOSA25 data analysis to be an upper limit of cluster sizes in high-resistivity sensors, and two encoded states per hit should therefore be a safe estimate. Later analysis with uniform cut of all pixels confirms this assumption, as a data analysis with the uniform cut of 120 electrons cut gave average multiplicity of 3.1 pixels and a slightly higher cut of 150 electrons gave an average multiplicity of 2.6, and without significant degradation in detection efficiency between the two cases (personal correspondance with R. De Masi).

For line widths of 1024 pixels, an encoded state contains 10 address bits and two encoding bits. Using one 16-bit word to store each 12-bit state, two states amount to two 16-bit words (when ignoring the line address header that precedes all the states in a line). For an encoded cluster, however, 10 bits are required for line address, 10 bits for column address, and 8 bits for encoding the cluster shape. Two 16-bit words are required for storing an encoded cluster, and the amount of required storage for an individual hit in a line therefore becomes the same.

With the small clusters assumed in high-resistivity substrates, clusterization increases the logic complexity without necessarily giving significant data reduction. The difference in data reduction is in the end dependent on both cluster multiplicity and hit density, as encoded states in a line also require the 16-bit line address header that contributes to the amount of data, while the two 16-bit words storing each cluster have sufficient space to embed the line address.

However, as soon as the average number of states per hit goes below two, the total amount of data will for a certain hit density become equal or smaller for state encoding than for clusterization. Assuming a system that is built to scale with high hit densities and with an average of two or fewer states per hit, the conclusion on data reduction is therefore that we continue with implementing state encoding, and not clusterization.

5.1.5 Proposed architectures

Combining new technology like 3D integration and sensors with fully depleted epitaxial layer opens the perspective of running the CBM at a collision rate of 10^6 collisions/second.

A conceptual design for a detector, based on these features, is therefore to be presented. It is a four-tier architecture with a dedicated layer for detector, a separate layer for analog signal processing and two layers for high performance digital readout.

Detector tier: XFAB 0.6 high-resistivity epitaxial process (XC06) [1], bonded to amplifier tier with Ziptronix DBI
Analog amplifier tier: Tezzaron/Chartered
Digital tiers: Tezzaron/Chartered

As a starting point, one assumes a pixel matrix of approximately the same dimensions as for MIMOSIS-1 [63], but with a rolling shutter where it is assumed that the line processing time is reduced down to 40 ns:

- Pixel matrix of 500 rows x 1024 columns
- The 3D integrated rolling shutter chip can be divided into submatrices to meet the timing requirements.
- For 2 µs time resolution, the number of lines to be processed during one frame is limited to $\frac{2\,\mu\mathrm{s}/frame}{40\,\mathrm{ns}/line}$ = 50 lines/frame, and this determines the number of lines in each submatrix.
- A pixel sensor of 500 lines is therefore organized in 10 submatrices read in parallel. See Figure 5.4.

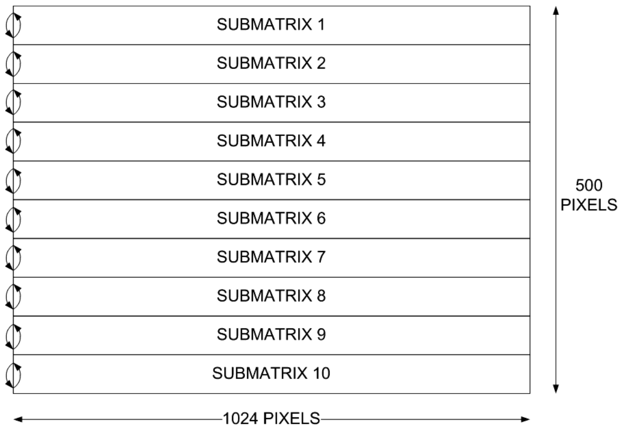


Figure 5.4: Rolling shutter operated pixel matrix organized in 10 submatrices.

There are two different approaches for interfacing the segmented rolling shutter:

[1]Eventually XFAB 0.35 high-resistivity epitaxial opto-process (XO035) or even smaller high-resistivity epitaxial process when becoming available, as smaller process increases the tolerance of ionizing radiation.

- The first and most simple approach is to take the peripheral readout electronics of MIMOSA26 and put on top of the pixel matrix, using 3D integration to remove the inactive surface of the periphery. For a CBM application, this architecture has to be optimized to meet the requirements of four times smaller line processing time (down from 160 ns/line to 40 ns/line) and also a larger amount of states per line to process.

- The second approach is to design a completely new architecture that fully utilizes the new degrees of freedom given by 3D integration.

In this chapter, we follow the last approach, and a data-driven readout architecture with token passing and distributed state encoding in all the pixel rows has been designed. Individual pixels have access to the hit information of their nearest neigbours, and they are able to initiate their own readout by grabbing a token if they identify themselves as central pixels of valid states.

To provide a reference for the new architecture with respect to area, power consumption and scalability, the MIMOSA26 with its sparse scanned state encoding, multiplexer and memory management, was laid out in Charted-0.13 process, and the architectural changes required for a CBM application are also to be discussed later in this chapter.

To be compliant with the rolling shutter operation of the analog tier, the proposed architecture operates by the following principles:

1. It injects hits sequentially in the rows of one half of a submatrix (40ns/line).

2. It injects readout tokens in the rows parallel in the other half a submatrix (1us/line).

For the same frame readout time of 2 us, the critical time per line is thus increased from 40 ns to 1 us. See Figure See Figure 5.5.

Pixels grabbing the tokens are placing their address and the state encoding on a shared line bus, with line buffers storing the state information. These line buffers are then read out as a pipeline of scan flip flops [2].

[2]In the preliminary layout, the scan flip flops were implemented as standard D flip flops with multiplexed inputs

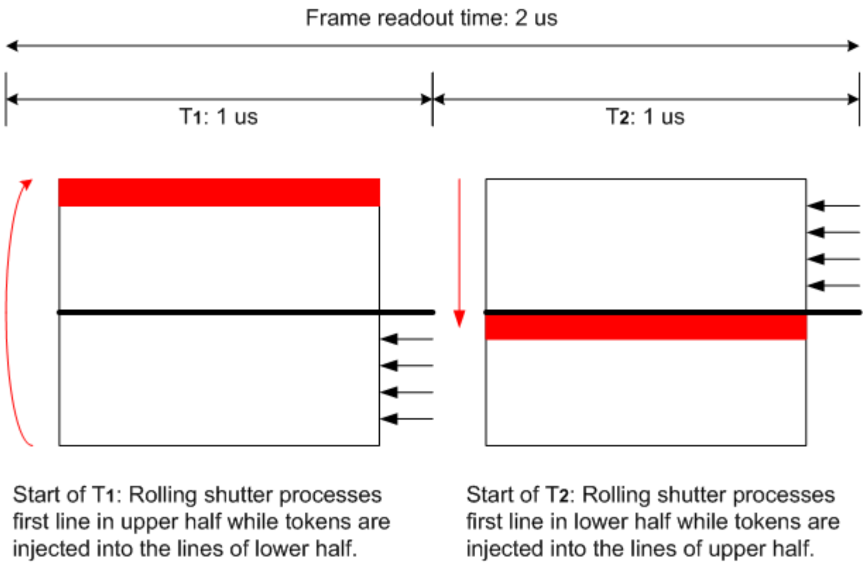

Frame readout time: 2 us
T1: 1 us
T2: 1 us
Start of T1: Rolling shutter processes first line in upper half while tokens are injected into the lines of lower half.
Start of T2: Rolling shutter processes first line in lower half while tokens are injected into the lines of upper half.

Figure 5.5: Splitting of submatrix in two halves to make token injection compliant with rolling shutter operation.

5.2 Data driven readout with tokens and distributed state encoding

5.2.1 Distributed state encoding and token passing between pixels in each line

To improve timing margins, we want to parallelize in all rows the zero suppression previously performed sequentially in the SUZE circuitry of MIMOSA26. For this purpose, we propose to equip the individuals pixels with autonomous state identification and encoding functionality. Figure 5.6 illustrates how the centralised state encoding of previous MIMOSA chips can be distributed into the individual pixels. Each pixel communicates with its nearest neighbours to identify and encode states.

A schematic of a pixel cell, with token passing circuitry and distributed state encoding, is shown in Figure 5.7 and Figure 5.8. Pixels identifying themselves as central pixels in a state may grab a token to transfer hit information. The logical blocks for the token passing are exactly the same kind of improved NAND/NOR cells that were already described in Chapter 3 and shown in Figure 3.31. After the proposed improvement, these blocks are small and compact enough to be put into

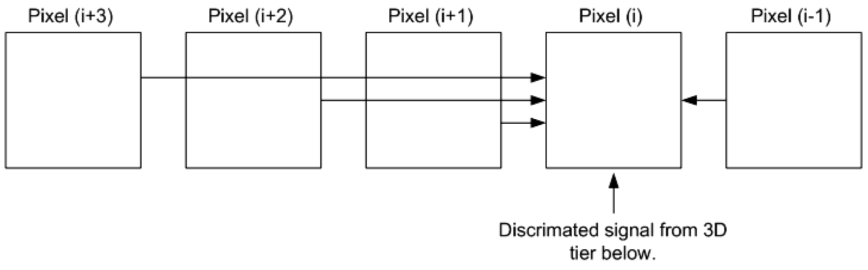


Figure 5.6: Operation principle of distributed state encoding. Every pixel in a line (pixel i) communicates with its three preceding pixels (pixel (i+3), pixel (i+2), pixel (i+1)) to perform the state encoding. It also receives an enable signal from its first succeeding pixel (pixel (i-1)), allowing the pixel to grab the flowing token if not already done by the succeeding pixel.

each cell in the pixel matrix.

The simplest condition for a pixel to identify itself as a state is that its own discriminator signal is one and the discriminator signal of its leftmost neighbor is zero. However, to provide a lossless and nonredundant encoding also for cases where there are patterns of ones longer than four, this hit identification logic has been slightly modified. In the final implementation, described more detailed in Appendix F, the pixels that see that its two preceding pixels all contain ones will propagate a signal to the third preceding pixel, demanding this third pixel to enable hit identification in the next pixel.

Every line has a column address bus and a state encoding bus for transferring the hit information to the end of line buffers. The pixels grabbing the token are mastering the two buses. The outputs of each stage are inverted to minimize delay through the chain of multiplexers, and each cell therefore provides a zero bit in addition to its address bits for making the logic in the receiving end capable to distinguish between data that has passed through an odd or even number of inversion stages.

Since the addresses of each pixel cell is fixed, the propagation path through the address bus is minimized even more by replacing the inverting multiplexer cells with NAND and NOR gates. For the address bits containing ones, the block_token signal is OR'ed together with the incoming address bit to provide the outcoming address bit. As long as the pixel cell is not blocking the token, the incoming address bit is therefore allowed to pass. When the pixel cell blocks the token, it will always override the incoming address with its own address bit of one value. Similarly, for zeroes, the active high block_token signal is fed through an inverter before beining NAND'ed together with the incoming address bit to provide the outgoing address

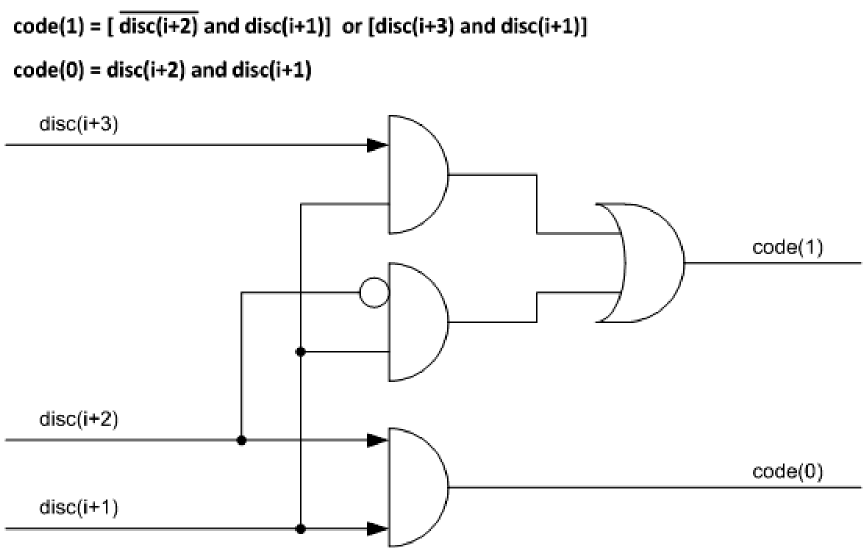

code(1) = [$\overline{\text{disc(i+2)}}$ and disc(i+1)] or [disc(i+3) and disc(i+1)]
code(0) = disc(i+2) and disc(i+1)
disc(i+3)
code(1)
disc(i+2)
code(0)
disc(i+1)

Figure 5.7: State encoding circuitry of a pixel cell.

bit. See Figure 5.9.

While the token and the addresses therefore fly through a fast chain of NAND and NOR gates, the slowest path through the line is the path of inverting multiplexers, necessary for propagating the code information.

With the above mentioned structures, an abstract view of a pixel cell with all its pinning will look like the one shown in Figure 5.10.

Such a pixel cell has already been successfully placed and routed on two different pitches, 18.45 µm and 14.76 µm, respectively. Three metal layers were then required for completing the routing. The netlist of the pixel cell was optimized in Synopsys Design Vision, where the appropriate constraints were set on desired area, token clock frequency, and capacitive load.

Due to the rolling shutter operation of the sensor, the tokens must be injected into the upper half while hits are injected into the lower half, and vice versa. A common state machine is responsible for reading out the upper and lower half of the matrix and to provide the two halves with their appropriate clocks and control signals.

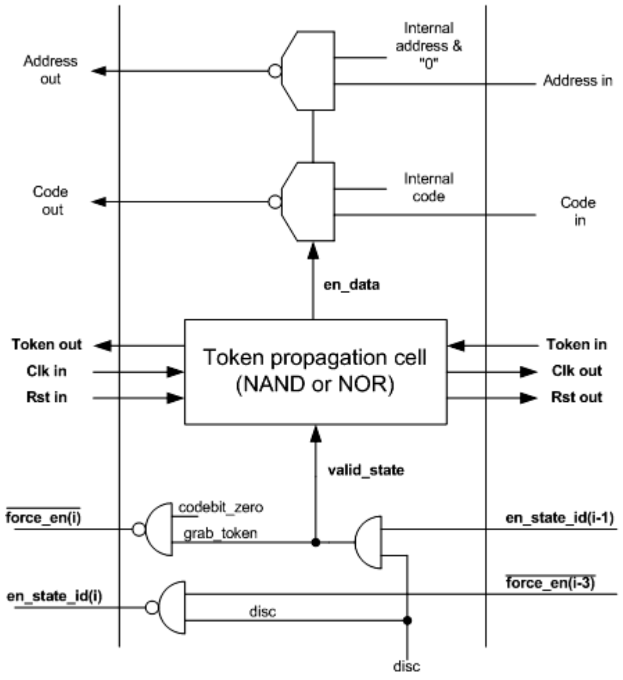


Figure 5.8: Circuitry for valid state identification and token-based data transmission in a pixel cell.

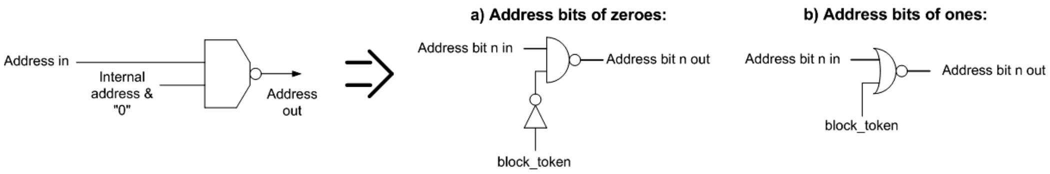


Figure 5.9: Since the address bits are fixed, the address multiplexer can be replaced by faster NAND and NOR gates.

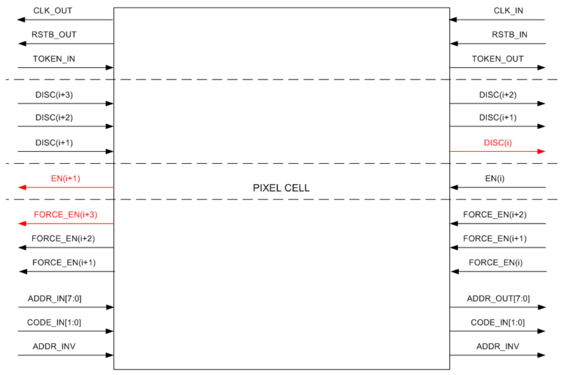


Figure 5.10: Abstract view of pixel cell.

5.2.2 Line segment structure

Partitioning

With 2 µs frame readout time, there is a time slot of 1 µs for the token to pass through all the pixels containing states in a line. To provide good timing margins, and scalability for a larger number of line buffers or even smaller frame readout times, it is proposed to split every line into 4 line segments. A separate token is being injected into every line segment. At the end of each segment, there are buffers to store the hit information placed on the data bus.

This structure has features similar to the end-of-column buffers of ATLAS FE-I3 [43], adopted to 3D technology. Where the ATLAS FE-I3 has the end-of-column buffers in the periphery, we will thanks to 3D integration be able to put end-of-line buffers on top to avoid dead area. See Figure 5.11.

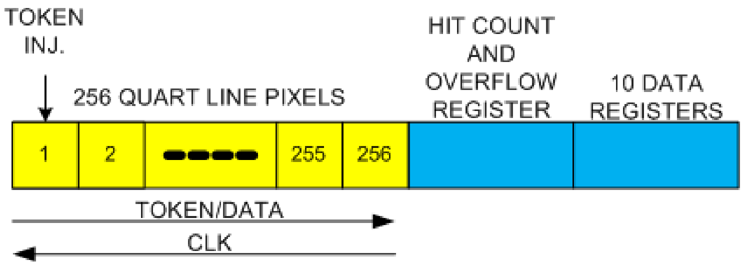


Figure 5.11: Organization of line buffers. Yellow: First digital tier. Blue: Second digital tier.

To determine the number of buffers per line segment, it is necessary to set the maximally tolerated number of hits per line segment.

For a given average number of hits, and with a given number of hits that can be stored in buffers, the cumulative Poisson distribution function gives the probability of overflowing the buffers and missing hits. We want to arrive at the number of buffers per line segment that corresponds to a probability less than 10^{-3} for missing hits in a line. With a frame readout time 2 µs, we calculate 500.000 frames/second, and with a collision rate of 10^6 collisions/second, the simulations show a worst case hotspot close to the beam hole with 7.2 $particles/mm^2/frame$.

Although different chip orientations could be considered to reduce the average hit density throughout a pixel line, the simplest and safest approach is to assume this worst-case hit density throughout an entire line.

Assuming pixel pitch of 18.45 µm and a line segment width of 256 pixels, the line segment size becomes:

$((18.45\,\mu\mathrm{m}) \cdot (18.45\,\mu\mathrm{m}) \cdot 256) = 0.0871\ mm^2.$

Assuming the worst-case hit density of 7.2 $particles/mm^2/frame$, we find the average number of hits in a segment to be:

$hit\_density \cdot line\_segment\_area = 7.2\ hits/mm^2/frame \cdot 0.0871\ mm^2/segment$
$= 0.627$ hits/segment/frame.

With such a worst-case density in all four line segments, the cumulative distribution function shows that up to five hits must be stored in each line segment to arrive at a probability of missing hits that for the entire line is below 10^{-3}.

Assuming a cluster multiplicity that corresponds to 2 states per hit, 10 buffers are then required for each line segment.

Timing performance

Since the token and address propagation go through a fast chain of NAND and NOR gates, the slowest path through the chain of cells can be assumed to be the propagation of code information through inverted multiplexers.

The worst-case situation would then be that there is a valid state in the first and the fourth rightmost pixels in a line segment. In this case the token clock first has to go through all the 256 pixels from the left to the right, activating the mostright pixel. After being released in the rightmost pixel, the token first flies through four pixel elements before being blocked in the fourth rightmost pixel. Here the code information is put on the bus and propagated through all the other 252 pixel cells before being sampled by the hit counter. See Figure 5.12. The minimum clock period would therefore becomes:

$$token\_clk\_delay \cdot 256 + token\_cell\_delay \cdot 4 + multiplexer\_cell\_delay \cdot 252$$

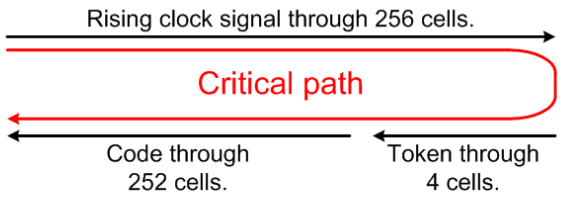


Figure 5.12: Critical path of a line segment of pixels.

As seen from the formula, the delay is decomposed into the following components:

Token clock delay

From the datasheet of the Artisan standard cell library, a minimum size clock buffer (3.69 µm x 1.84 µm) has an intrinsic rising delay of 0.075 ns plus an output load dependent rising delay of 3.24 ns/pF. The clock sees the 1.5 fF input capacitance of a DFFR flip-flop and the 1.2 fF input capacitance of the next clock buffer, in addition to the wire capacitance.

Total token clock delay is therefore:

$$0.075ns + 3.24ns/pF \cdot (1.5fF + 1.2fF + wire\_cap\_clk)$$
$$= 0.084ns + 3.24ns/pF \cdot wire\_cap\_clk.$$

Token cell delay

For the unblocked token flow, the token enters the NAND gates as one and enters the NOR gates as zero. The average propagation delay through one cell is therefore the transition from high to low in the NAND and the transition from low to high in the NOR, divided by two. With minimum size NAND and NOR gates, the NAND gate has intrinsic falling delay of 22 ps and load-dependent falling delay of 4.76 ns/pF, while the NOR gate has intrinsic rising delay of 33 ps and load-dependent rising delay of 9.70 ns/pF.

The NAND gate sees the inputs of one inverter (1.9 fF) and two NOR (2 x 2.4 fF), while the NOR gates sees the inputs of one NAND (2.4 fF), one inverter (1.9 fF), and one DFFR D-input (1.2 fF). Total load is therefore 6.7 fF for the NAND gate and 5.5 fF for the NOR gate. In addition comes the wire capacitance.

The total token cell delay therefore becomes:

$$\frac{[0.022ns+4.76ns/pF(6.7fF+wire\_cap\_nand)]+[0.033ns+9.79ns/pF(5.5fF+wire\_cap\_nor)]}{2}$$

$$= 0.070ns + 2.38ns/pF \cdot wire\_cap\_nand + 4.90ns/pF \cdot wire\_cap\_nor.$$

Multiplexer cell delay

Since the multiplexer is inverting, there will be a transition from high to low in half of the multiplexers and a transition from low to high in the other half. This is an improvement compared to a noninverting chain of multiplexers, as we avoid a worst-case scenario where the slowest type of transition is propagated through the entire path.

If the multiplexer is implemented as a minimum-size AOI22, it has an intrinsic rising delay of 60 ps and an intrinsic falling delay of 30 ps. It sees a 2.7 fF input capacitance of the next AOI22, and it has load-dependent rising and falling delays of 11.03 ns/pF and 4.67 ns/pF. The total multiplexer cell delay therefore becomes:

$$\frac{[0.06\ ns+11.03ns/pF\cdot(2.7fF+wire\_delay\_mux)]+[0.03\ ns+4.67ns/pF\cdot(2.7fF+wire\_delay\_mux)]}{2}$$

$$= 0.066\ ns + 7.85ns/pF \cdot wire\_delay\_mux.$$

Wire delay

For a rough estimation of the wire delay, we assume that the pixel cells are close enough to each others (18-20 µm) to make the gate delay dominant with respect to the wire delay, and the actual delay should therefore not be too far above these initial estimations. To obtain a safe worst-case value for the wire delay, however, an analog simulation was performed with a 0.2 µm wide and 20 µm long metal2 wire, surrounded by metal everywhere and with 0.1 µm spacing between the metal2 wires. Figure 5.13 shows the setup. From this simulation we extracted an AC capacitance of 5.6 fF.

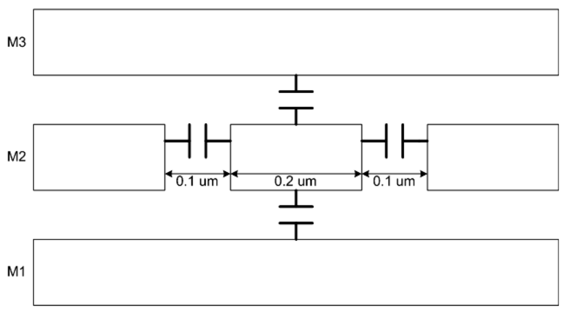


Figure 5.13: Cross-section of 20 µm long metal 2 wire, surrounded by metal everywhere at smallest possible spacing to provide worst-case reference for wire delay.

Total delay

Using the extracted worst-case value for all wire capacitances, the finally estimated delay values become:

Token clock delay: 0.10 ns.

Token cell delay: 0.11 ns.

Multiplexer cell delay: 0.11 ns.

Although the initial assumption was that the NAND/NOR chain would be the fastest path, and the multiplexer cell delay would be the slowest path, it turns out that the NAND/NOR chain and the multiplexer path actually may have equal delay if nothing is done to buffer or balance the larger fanout of the NAND and NOR gates. However, as the path through the NAND/NOR chain and the path through the multiplexers are approximately equal, one can as well turn the argument and say that the NAND/NOR path is already optimally balanced, as an improvement of the speed through the NAND/NOR chain will not improve the speed of the critical path.

With these numbers inserted, the critical delay becomes:

$$token\_clk\_delay \cdot 256 + token\_cell\_delay \cdot 4 + multiplexer\_cell\_delay \cdot 252$$
$$= 0.10 \cdot 256 + 0.11 \cdot 4 + 0.11 \cdot 252 = 53.76\ ns.$$

With 10 line buffers, 10 token clock cycles are required. Dividing one microsecond by eight gives a minimum required clock period of 100 ns, and we therefore see that the timing requirements are met with very good margin.

Basic building blocks of buffers

For every token clock cycle, the content on the line data bus is stored into the leftmost buffer, with the content of all the buffers being shifted from one buffer to the next. After 10 token clocks, up to 10 valid states have been stored in each their buffer. A hit counter increments the number of valid states as long as the token injected into the line has not been returned. See Figure 5.14.

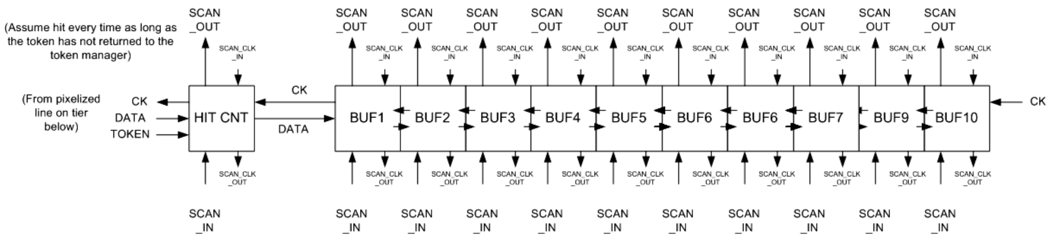

(Assume hit every time as long as the token has not returned to the token manager)
(From pixelized line on tier below)
CK
DATA
TOKEN
HIT CNT
CK
DATA
SCAN _OUT
SCAN_CLK _IN
SCAN_CLK _OUT
SCAN _IN
BUF1
BUF2
BUF3
BUF4
BUF5
BUF6
BUF6
BUF7
BUF9
BUF10
CK

Figure 5.14: The line buffers can be operated in acquisition mode and in scan mode.

More detailed schematics of line buffer registers and the line buffer counter can be found in Appendix F.

5.2.3 Readout

Modes of operation

The column buffers are implemented as scan flip-flops, and their content is shifted from line to line during readout. See Figure 5.15. As the height of the column buffers is the same as the height of the pixel cells, 18.45 µm, it would be impossible to implement a mux for the same purpose, due to routing congestion.

A finite state machine (FSM) controls the readout and provides the scan clock to the line buffers. To ask for a new line and then safely latch the content of the new line takes two clock periods. If the number of states in a line is larger than the number of states that are selected for each clock cycle, then the state machine can calculate exactly when to request a new line and have the new line latched exactly at the clock cycle when the last states in the current line are selected.

However, there are also situations where the number of states in a line is lower or equal to the number of states that are selected during one clock cycle. To avoid waiting cycles during the request of a new line, the pipeline extends two stages into the clock domain of the FSM, allowing the FSM to read one state and store this state together with its line address while waiting for the pipeline to advance.

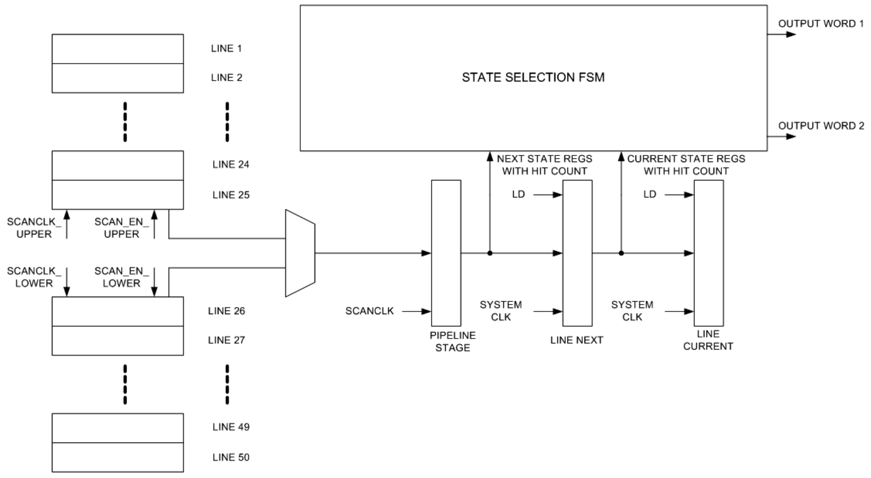


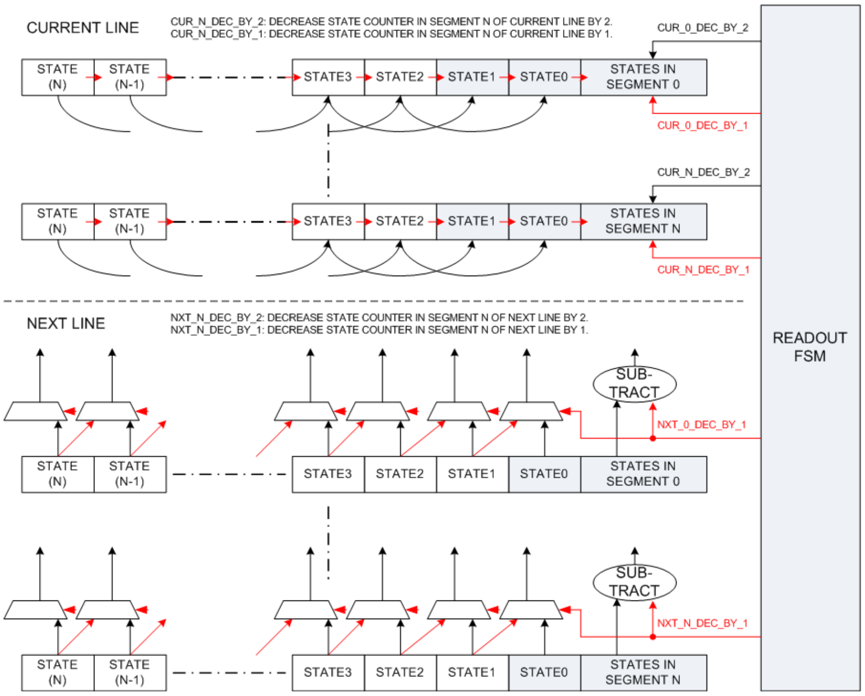
Figure 5.15: Readout of upper and lower half of a submatrix, controlled by a common FSM. The FSM performs pipelined readout of the buffers in one half while the token passing circuitry provides new content to the buffers in the other half.

State selection mechanism

With four line segments, each composed of 10 line buffers, a total of 40 buffers are to be examined by the readout FSM. A multiplexer proportional to the total number of buffers is thus implied, unless measures are taken to avoid such a structure. By extending the scan chain into the pipeline of the readout FSM, it is possible to implement a structure where the number of buffers seen by the state machine is independent of the number of buffers inside each line segment. Its concept is illustrated in Figure 5.16.

The readout FSM starts by asynchronously scanning through the state counter registers of the different segments, selecting the two first segments that contain non-zero hit information. The state machine then selects the two first states of the first selected segment as long as the number of states in this segment is larger than one. Otherwise it selects one state from the first selected segment and one more state from the second selected segment.

If two states are extracted from the first segment, then the state counter register of the first segment is decremented by two, and the content of the first buffer is shifted two places. Otherwise, if only one state is extracted from each of the two selected segments, then the state counter registers of both the two segments are each decremented by one and the content in each of the two buffers is shifted one place.



CURRENT LINE
CUR_N_DEC_BY_2: DECREASE STATE COUNTER IN SEGMENT N OF CURRENT LINE BY 2.
CUR_N_DEC_BY_1: DECREASE STATE COUNTER IN SEGMENT N OF CURRENT LINE BY 1.
CUR_0_DEC_BY_2
STATE (N)
STATE (N-1)
STATE3
STATE2
STATE1
STATE0
STATES IN SEGMENT 0
CUR_0_DEC_BY_1
CUR_N_DEC_BY_2
STATES IN SEGMENT N
CUR_N_DEC_BY_1
NEXT LINE
NXT_N_DEC_BY_2: DECREASE STATE COUNTER IN SEGMENT N OF NEXT LINE BY 2.
NXT_N_DEC_BY_1: DECREASE STATE COUNTER IN SEGMENT N OF NEXT LINE BY 1.
SUB-TRACT
NXT_0_DEC_BY_1
NXT_N_DEC_BY_1
READOUT FSM

Figure 5.16: State selection mechanism of the readout FSM. For each clock cycle, the state counters of the selected segments are decremented and the content of the buffers is shifted one or two places. The buffers that are seen by the readout FSM are shown in gray.

After decrementing the state counters and shifting the content of the line buffers, the procedure can be repeated until all the state counters are equal to zero and a new line is demanded. The scan for states to read out can therefore always start with the first line segment.

With this structure, the number of buffers seen by the state machine also becomes equal to 4 x 2 = 8, instead of 4 x 10 = 40.

As one word is read from the next line while waiting for the pipeline to advance, a similar structure is also implemented for the next line, with the selected hit counter being subtracted by one before being loaded into the next stage, and with the content of the selected segment being shifted one place.

Compared to the multiplexer structure of MIMOSA26, the new state selection FSM has the following improvements:

- Its size is independent on the number of buffers in the line segments.
- Decrementing the state counters for each clock cycle a segment is read, and allowing the scan for states to always start from the rightmost segment, there is no need for a token-passing loop that otherwise consumes two clock periods for each time states are selected.

Operating frequency

The operation frequency of the readout FSM depends on the number of states to read and the number of states selected per clock cycle.

When the work on the conceptual design started, the simulation results were not yet ready. To have some margin, an average hit density of $3.5 \cdot 10^6 hits/mm^2/second$ ($7.0 hits/mm^2/frame$) was assumed.

Also assuming a pixel pitch of 18 µm and a matrix size of 512 rows and 1024 columns, the matrix size in mm^2 became:

$$\frac{((18\cdot10^{-6}\text{m})\cdot(18\cdot10^{-6}\text{m})\cdot1024\cdot512)}{10^{-6}m^2/mm^2} = 169.87\ mm^2.$$

3.5 million $particles/mm^2/sec$ then gives:

$$hit\_density \cdot matrix\_area \cdot frame\_readout\_time$$
$$= (3.5 \cdot 10^6 hits/mm^2/second) \cdot (169.87 mm^2) \cdot (2 \cdot 10^{-6}\text{µs}) = 1190 \text{ hits/chip/frame}.$$

For a half submatrix, this number translates into an average of 59.5 hits. Which for a hit miss probability less than 10^{-3} corresponds to reading out up to 85 hits, which again translates into reading 170 states. In addition comes 25 line addresses, and thus in total 195 words to process during 1 µs. If two words are selected at each clock cycle, then the clock frequency of the FSM can be set to 100 MHz. While with four words being selected, the clock frequency can be relaxed to 50 MHz.

After the simulation results were made available (obtained from private correspondance with Michael Deveaux), one sees that the worst-case arrangement of a chip gives an average number of 3.4 $hits/mm^2/frame$.

3.4 $hits/mm^2/frame$, in a half submatrix of $25 \cdot 4 = 100$ line segments, then gives:

$$\begin{gathered} hit\_density \cdot matrix\_area \\ = 3.4\, particles/mm^2/frame \cdot 0.0871\ mm^2/segment \cdot 4\ segments \cdot 500\ lines \\ = 583.60 \text{ hits/matrix/frame} = 29.2 \text{ hits/half-submatrix/frame.} \end{gathered}$$

To provide a hit miss probability less than 10^{-3}, the number of hits is expanded to 47, which in the next turn translates into 94 states. With 94 states and 25 line addresses, where each state and each address accounts for one word, a total of 119 words are thus to be stored during one microsecond. One therefore sees that 100 MHz FSM operation with selection of two words at a time gives very good margins for storing all the hits in a submatrix.

A relatively high clock frequency may be allowed for the FSM, as this frequency does not affect the number of clock pulses given to the flip flops of the line buffers, which will always remain at 25 per submatrix during 2 µs.

5.2.4 Floorplanning and layout

An overview of the line buffer organization in a submatrix is shown in Figure 5.17. By varying the direction of the token, one has some degree of freedom on where to locate the buffers. With the location of the column buffers put into place, it is possible to locate the other necessary modules, as shown in Figure 5.18.

A 3D stack for the entire architecture is shown in Figure 5.19, with the digital pixel in one tier, and the line buffer, readout FSM and memory together in another tier.

5.2.5 Serial output links

The typical delay for a standard I/O cell in Chartered technology is 2 ns (500 MHz). With 8b10b serial transmission at this speed, 50 bytes (25 words) can be transferred serially during 1 µs.

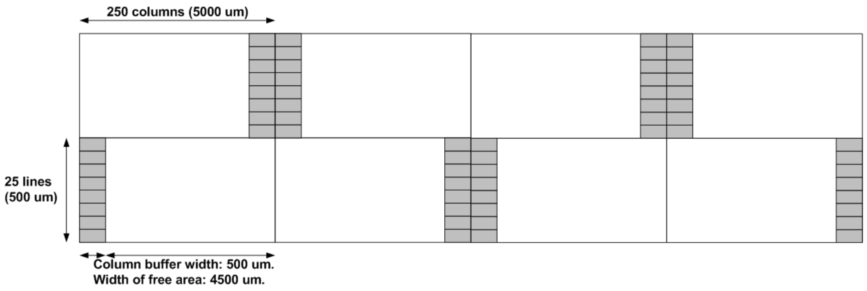


Figure 5.17: Floorplan of the second digital tier with the column buffers allocated (top view).

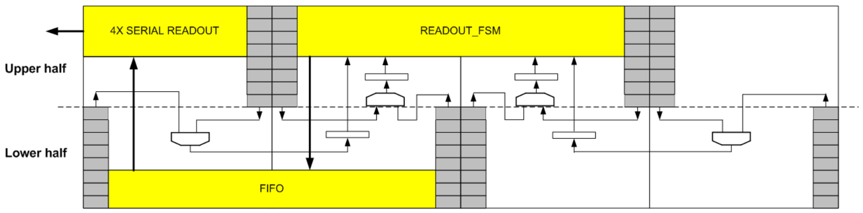


Figure 5.18: Possible floorplan of second digital tier with the column buffers and readout structures allocated.

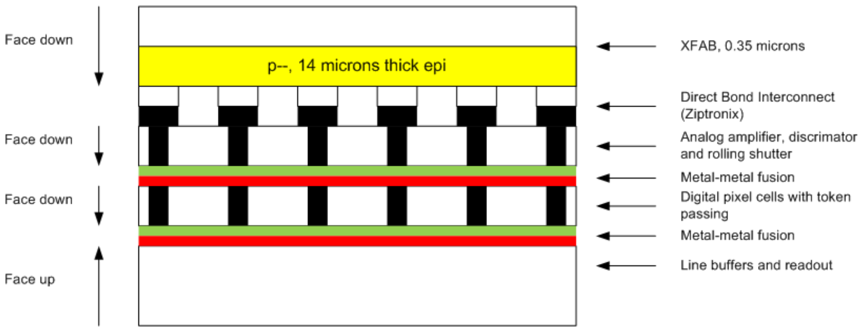


Figure 5.19: 3D stack of the entire architecture.

With the worst-case orientation of a chip, it was shown earlier in this chapter that there will be an average of 583.6 hits/chip/frame. If all the submatrices are interfaced by a common infinitely deep FIFO, the data rate can in theory be relaxed down to a level corresponding to the average hit arrival rate. 584 hits/chip/frame translate into $584 \cdot 2$ states and 500 line addresses, in total 1668 words. As 50 words can be serial transmitted during one frame, the theoretical minimum number of serial links becomes $\frac{1668}{50} = 34$ serial links.

With the theoretical minimum number of serial links per chip being 34, the actual number of serial links required for data transmission is dependent on the buffering and transmission scheme:

First approach: Direct transmission

With direct transmission, the states selected by the readout FSM are transferred directly out of the chip, without any kind of buffering.

With the initial assumption of $3.5 \cdot 10^6 hits/mm^2/second$, the required number of serial links for this transmission scheme becomes $\frac{195\ words}{25\ words/link} = 8$ serial links.

8 serial links are thus required for one submatrix, with 80 in total for the entire chip.

While with the numbers obtained from physics simulation, and the worst-case orientation resulting in 119 words to transfer during one microsecond, the required number of serial links becomes $\frac{119\ words}{25\ words/link} = 5$ serial links.

5 serial links are thus required for one submatrix, which in total becomes 50 serial links for the entire chip.

However, sending 5 bytes in parallell is inconvenient when interfacing groups of 16-bit words, as the last word must be cut in two and the next half sent during the next transmission.

Second approach: With ping-pong buffer in every submatrix

In the entire submatrix, the average number of hits becomes 60, which when providing hit miss probability less than 10^{-3} is expanded to 85 hits, and therefore 170 states. These states, together with 50 line addresses, must be transferred during the 2 µs frame readout time. Two memories of minimum size $170\ words + 50\ words = 220\ words$ are thus required, and the required number of serial links then becomes $\frac{(170+50)\ words}{50\ words/link} \approx 5$.

One can therefore conclude that simple ping-ping buffering does not give any advantage in terms of serial links, although it has the benefit of continuously transferring only valid data until the entire half-submatrix is read out - while direct transmission will have the situation of transferring idle states when no valid data is available from the pixel lines.

Third approach: Implement a properly dimensioned FIFO using queue theory to obtain 4 links per submatrix

The theoretical minimum number of serial readout links for the entire chip is 34. However, if each submatrix is to have independent readout, the theoretical minimum number of serial links per submatrix must be four when utilizing a sufficiently deep FIFO.

To obtain the required FIFO depth, we start by modeling the readout as an M/D/1 queue (see for example [57]), with the the incoming hits being service requests and the serial links are considered as one single server.

The long-term utilization of the server is written as:

$$\rho = \frac{\lambda}{\mu}$$

Here is λ the average number of hits arriving in the submatrix, and μ is the hit processing rate. We therefore find that λ must be 60. To obtain μ, we know that each serial link can transmit 50 words during one frame, and with two states per hit and 50 line addresses per submatrix, the rate of hits transferred through one link becomes:

$$hits/link = \frac{words/link}{words/hit} = \frac{50}{(60 \cdot 2 + 50)/(60)} = 17.7.$$

With m serial links, the hit processing rate becomes $\mu = m \cdot 17.7$, where m is the number of serial links. The utilization of the server then becomes $\rho = \frac{60}{m \cdot 17.7} = \frac{3.4}{m}$, and we therefore see that 4 serial links are necessary to have a utilization below 100%.

For an M/D/1 queue, the probability of having exactly j customers in the queue becomes:

$$P_j = (1-\rho) \sum_{i=0}^{j} e^{i\rho} (-1)^{j-i} \frac{(i\rho + j - i)(i\rho)^{j-i-1}}{(j-i)!}$$

And the cumulative probability of having more than j customers in the queue becomes:

$$Q_j = 1 - \sum_{i=0}^{j} P_i$$

With a utilization of $\rho = \frac{3.4}{4} = 0.85$, the cumulative distribution function gives a probability less than 10^{-3} when the number of customers in the queue is 22 or larger.

With an average of 60 hits, corresponding to 120 states and 50 line addresses, the average number of stored words per hit becomes $\frac{170}{60} = 2.83$. The required FIFO length in terms of words therefore becomes $22 \cdot 2.83 \approx 62$.

With such a small buffer size, it could be more efficient in terms of area to use a register file instead of RAM for implementing the FIFO.

Conclusion on buffering

Direct transmission requires 5 serial links per submatrix, in total 50 serial links. In case of empty lines, there will be stall cycles and therefore idle words transmitted in the middle of the data stream.

Buffering with a ping-pong memory should in theory reduce the number of serial links, but it turns out to become the same as for direct transmission. 5 serial links is still an inconvenient number, as the last 16-bit word transferred must be cut in half and sent in next transmission.

Replacing the ping-pong buffer with a properly sized FIFO reduces the number of serial links per submatrix from 5 to 4, which is theoretical minimum number of serial links, unless a common serial readout interface for all the submatrices is implemented. The calculations also show that the amount of required storage is reduced from two memories of minimum 220 words to one FIFO of 62 words, and thus small enough to be implemented without using RAM at all.

Choosing a properly sized FIFO for the data transmission therefore minimizes both memory usage and the number of serial transmission links.

5.2.6 Noisy pixel contribution to data stream

As already stated in Chapter 1, laboratory tests on high-resistivity MAPS show that the SNR of these pixels is twice as good as that of standard MAPS, which means that fake hit rates well below 10^{-5} could be expected prior to irradiation. With a fake hit rate of 10^{-5}, there will in a CBM MVD matrix of 500,000 pixels be approximately five fired pixels in total and therefore negligible as it is just a small fraction of the Poisson distributed fluctuations in hits due to particles (an average of around 580 hits/chip/frame due to particles in the worst-oriented chip, as previously stated in this chapter).

Still, there are other factors that complicates such an assumption:

- The excellent SNR features of MIMOSA25 were confirmed even after a non-ionizing radiation fluence of $3 \cdot 10^{13} neq/cm^2$, but it remains to verify that the SNR is not significantly degraded when fluences increase to the order of $10^{14} neq/cm^2$.

- Although the transistors in 0.13 µm Chartered/Tezzaron designs have excellent post-irradiation performance even after hundreds of megarads, the same excellent performance cannot be expected in the detector tier where the charge collecting diode and the preamplifier are implemented in an XFAB process with the larger 0.6 µm or 0.35 µm feature size. Also here it remains to perform the relevant tests.

- As already mentioned in Chapter 2, test results of MIMOSA26 at frame readout times of 90 and 115 µs show a dependency between fake hit rate and frame readout time. And although it remains to be further investigated, it is not unreasonable to assume that a reduction in frame readout rate down to 2µs helps to further reduce the fake hit rate.

Due to the lack of relevant tests results, there are currently no realistic estimates on the expected fake hit rate of a high-resistivity epitaxial MAPS based sensor installed in the CBM MVD. For the calculations performed in this chapter, the contribution of the fake hits was considered negligible (fake hit rate $< 10^{-5}$) and therefore not necessary to take into consideration. But as soon as the relevant test results from prototypes are available, one should if necessary redo the hit density calculations with noisy pixels taken into consideration, and adjust the size of memories for adapting them to a higher data rate due to noisy pixels.

5.2.7 Power consumption

As long as there are no hits in the pixel cells, there are no signals switching in the pixel matrix, except the token signal switching one time and 10 token clocks sent to the token flip flops during 2 µs. For the line buffers, there are 10 token clocks and 25 scan clocks are provided during 2 µs. We therefore find that we have 5 MHz clock frequency for the pixel cells and 17.5 MHz for the buffers. The low frequencies should imply low dynamic power consumption.

Two different power estimations have been performed. The first was run in the Encounter layout tool with the expected clock frequency and average switching rate as input parameters. The second one included the toggling statistics obtained from simulation.

Estimations from Encounter, power analysis based on assumed switching factor of 0.2

Pixel cells, NAND: $(1.4 \cdot 10^{-4}\ mW) \cdot 500\ rows \cdot 512\ NAND\ columns = 35.8\ mW$
Pixel cells, NOR: $(1.5 \cdot 10^{-4}\ mW) \cdot 500\ rows \cdot 512\ NOR\ columns = 38.4\ mW$
Line buffer registers: $(24.5 \cdot 10^{-4}\ mW) \cdot 4 \cdot 10 \cdot 500 = 49.0\ mW$
Line buffer counters: $(10.4 \cdot 10^{-4}\ mW) \cdot 4 \cdot 500 = 2.1\ mW$

Total power consumption: 125.3 mW.

In addition to the pixels and the buffers comes the power consumption of the readout FSM and the serial links. Here the Encounter power analyzer showed that the readout FSM had an estimated power of 0.74 mW while the four serial readout links has an estimated power of $4 \cdot 0.74\ mW = 2.96$ mW, in total 3.7 mW. We therefore see that the power contribution of these circuits are negligible compared to the power of the pixels and the line buffers.

Estimations from Encounter, power analysis based on VCD files from simulation

Pixel cells, NAND: $(2.0 \cdot 10^{-4}\ mW) \cdot 500\ rows \cdot 512$ NAND columns = 51.2 mW.
Pixel cells, NOR: $(1.6 \cdot 10^{-4}\ mW) \cdot 500\ rows \cdot 512$ NOR columns = 41.0 mW.
Line buffer registers: $(89.0 \cdot 10^{-4}) \cdot 4 \cdot 10 \cdot 500\ rows = 178.0\ mW.$
Line buffer counters: $(32.1 \cdot 10^{-4}) \cdot 4 \cdot 500 = 6.4\ mW.$

Total power consumption: 276.6 mW

Estimations for memory

Selecting the optimum solution with a 62-word deep FIFO and four serial links, one can implement the FIFO with two 32-word deep register files, generated using the Artisan register file compiler. With the register file operated at 100 MHz frequency, one obtains:

Write AC current: 0.57 mA => $2 \cdot 1.5\ V \cdot 0.57\ mA = 1.71\ mW$.

Read AC current: 0.35 mA => $2 \cdot 1.5\ V \cdot 0.35\ mA = 1.05\ mW$

We see that with a properly dimensioned FIFO, the power consumption due to memory becomes negligible compared to the power consumption of the pixels and the line buffers.

Conclusion

Both estimates arrived at approximately similar results for the pixels cells, but with a power consumption for the line buffer registers and counters that in the last estimation is more than three times higher than in the first estimation. The reason for the low power in the first estimation can, however, be traced back to the fact that the clock in these registers is gated, and a power estimator based on low-activity, random inputs will therefore block the clock from arriving at almost half of the time.

Also rough estimations based on the datasheet verify that the estimations from simulation are the most correct. A DFFRX4 flip flop, even with no activity at its D input, consumes 0.04 µW/MHz. Containing 10 flip flops, switching at an average rate of 17.5 MHz, the power consumption of a buffer due to the clock input becomes $70 \cdot 10^{-4}$ mW, which is right below the results obtained from simulation based power analysis.

With the highest and most reliable result showing no more than 280 mW in total, the estimations imply low power consumption.

5.2.8 Scalability of architecture

In the proposed conceptual design, whose dimensions were set on the basis on the simulation results for CBM with a collision rate of 10^6 collisions/second, there are 4 segments per line and 10 buffers per segment. To adapt the design to even higher hit densities, one may increase the number of buffers per segment or one may increase the number of segments.

Reducing segment size and increasing number of buffers per segment

As increasing the number of buffers means increasing the token clock frequency, there will due to timing requirements be a maximum number of allowed buffers for a given segment size. Increasing the hit density beyond the range covered by the maximum number of buffers therefore also requires an increase in the number of line segments.

As the number of line segments grows larger, there will be more blockage in the layout and less space to allocate the other modules required for the readout (FSM and serial transmission circuitry), and at some point, a separate tier only for the line buffers is required. However, with a separate tier allocated only for the line buffers, one is ultimately allowed to increase the number of line buffers to an amount where all of the layout is occupied by the buffers.

Preliminary layout results show a line buffer width of 50 µm. As the width of the entire line is 84.45 µm x 1024 = 18.9 mm, the maximum number of line buffers becomes 377.

Starting with only 4 line segments of 256 pixels, it has been shown that the minimum token clock period could safely be reduced down to 55 ns. With one microsecond processing time on each of the half-submatrices, the theoretical upper limit of token clocks with this segment size therefore becomes $1000\ ns/55 \approx 18$, which gives a maximum number of buffers of 18. If the number of buffers exceeds this, the design starts to become marginal with respect to timing, unless for example look-ahead logic is inserted to increase the speed of token (which is fully possible, although it increases the amount of logic inside each pixel cell and thus may increase the minimum size of the pixel cell).

For the simplicity of comparing the different segment sizes and their number of line buffers, we assume one uniform average hit density over the entire line of four segments (instead of a more realistic model where the segment closest to the beampipe has a higher average density than the three other segments). It can then be shown that for a hit miss probability equal to 10^{-3} for the entire line, the hit miss probability in the four individual segments must be equal to $2.6 \cdot 10^{-4}$. With up to nine hits being stored in one segment, such a hit miss probability corresponds to an average of 2.9 hits per line segment, which in the next turn translates into an average of $\frac{2.9\ hits/segment/frame}{0.0871\ mm^2/segment} = 33.3\ hits/mm^2/frame$.

Similar calculations have been made for segment sizes of 128 and 64 pixels, and their results are depicted in Table 5.1.

One sees from the table that it should in principle be possible to scale the architecture into ranges of hit densities more than an order of magnitude above the simulated worst-case hit density of CBM MVD.

One also notes that with a line buffer width of 50 µm, obtained from preliminary layout results, 32 line buffers in each of the 8 segments means that almost 70 percent of the chip area is occupied by the line buffers. With careful organization of the layout, it should still be sufficient area for a readout FSM and serial transmission on the same tier. Further increasing the number of segments to 16, there can be up to 70 buffers per segment before the design becomes timing marginal - but with such a large amount of segments, there is only sufficient area to allocate 23 buffers per segment [3]

While the solution with the maximally allowed number of buffers with 8 segments could still fit into one tier, the maximally allowed number of buffers for 16 segment cannot be achieved, not even with a separate tier only for the line buffers. From the

[3]Matrix width of 1024 x 18.45 µm, divided by 50 µm, gives maximally 377 buffers, which divided on 16 segments becomes 23 buffers per segment.

table it is also clearly seen that with the area limitations, the number of available buffers in the 16-segment solution is too small to provide any improvement for the hit density.

Assuming the dynamic power consumption to be dominant, the power consumption of the different solutions is presented in normalized form, with the 4-segment, 10-buffer solution being the reference [4] [5].

Interfacing the increasing number of buffers

The problem with an ever-increasing multiplexer for selecting between the increasing number of buffers is solved by using the previously described state selection structure, where the FSM selects states from the end of a pipeline that is incremented for each clock cycle. The two parameters that define the number of buffers that are seen by the FSM are the number of segments and the number of states that are selected for each clock cycle. If the FSM is designed to select two states at each clock cycle, then it will even in the case of 16 segments (with in total 16 x 23 = 368 buffers) not see more than 16 x 2 = 32 buffers, and the implied multiplexer will still be of a moderate size.

[4] Normalized pixel power: $\frac{1}{10} \cdot N$, where N is the number of buffers per segment (equal number of token clocks per frame).

[5] Normalized line buffer power: $\frac{1}{4 \cdot 10 \cdot (25+10)} \cdot M \cdot N \cdot (25+N)$, where M is the number of segments

Table 5.1: The hit densities that can be covered are shown as function of line segments and line buffers.

Line segments	**4**	**4**	**8**	**16**	**16**
Line buffers	10	18	32	70	22
Miss probability/segment (Maximally allowed)	$2.6 \cdot 10^{-4}$	$2.6 \cdot 10^{-4}$	$1.3 \cdot 10^{-4}$	$6.5 \cdot 10^{-5}$	$6.5 \cdot 10^{-5}$
Average of hits/segment.	0.85	2.45	5.80	17.4	2.8
Average of hits/mm^2. (Maximally allowed)	9.6	28.1	133.1	799.1	128.2
Token clk freq.	5 MHz	9 MHz	16 MHz	35 MHz	11 MHz
Power consumption pixels	1	1.8	3.2	7	2.2
Power consumption buffers (Power is normalized)	1	2.2	10.4	76.0	11.8

For a given experiment, buffers and FSM should therefore be balanced according to the following principles:

- The line segments and the line buffers must be dimensioned to take into account the highest densities in certain lines or line segments.
- The readout FSM must be dimensioned to take into account the highest number of hits in the entire chip.

The possibility of making the readout independent on the number of states per line is a clear improvement compared to the sparse scan and multiplexer structure of SUZE.

Further reduction of frame readout rate

Instead of considering the time resolution fixed, and increasing the number of buffers and segments to improve hit density, one can go the other way by considering a fixed worst-case hit flux (which according to the CBM simulation results should be $3.6 \cdot 10^6$ $hits/sec/mm^2$), operate the token passing circuitry at the maximally allowed frequency and set the number of line buffers to the minimally required for covering the worst case hit density through the entire line.

One then finds that with 4 line segments, it is sufficient with six buffers in each segment when these are operated at the maximal 55 ns token clock frequency and therefore imposing a frame readout time of 660 ns.

Still considering the same worst-case orientation of the sensor as before, the average number of hits in a submatrix reduces from 30 to 10. To obtain a hit miss probability less than 10^{-3}, this number of hits expands to 21. With 42 encoded states and 25 line addresses, the number of words to store during a half frame becomes 67, and the readout FSM must therefore be operated at 200 MHz frequency.

With similar calculations for 8 segments, one arrives at four buffers, 220 ns frame readout time, and a readout FSM operating frequency of 410 MHz. While for 16 segments, one arrives at two buffers, 110 ns frame readout time and 710 MHz operating frequency.

A frame readout time of 110 ns is at the same order of magnitude as the charge collection time in standard MAPS (although still far above the charge collection time achievable in depleted epitaxial MAPS, which is less than a few nanoseconds). However, with 50 lines in each submatrix, a line processing time of 2.2 ns is required for the analog rolling shutter circuitry - more than an order of magnitude better than the expected performance for the first 3D integrated rolling shutter circuits. One could imagine that the frame readout times of 80 or 160 ns are achieved by increasing the number of submatrices to 250 or 125, but such an approach would explode the power budget.

Conclusion is therefore that the analog rolling shutter circuitry remains the limiting part in terms of improving time resolution, while the proposed digital readout can be adapted to all the fastest rolling shutter circuits that are expected to be designed in the near future.

5.3 SUZE on top of the rolling shutter

The rolling shutter has discriminator in every pixel, not a shared discriminator for each column. If the sensor matrix is to be processed linewise, a shared discriminator bus for each column must be implemented. The discriminator column bus should preferentially be implemented in the rolling shutter chip, otherwise a separate tier must be used for this purpose.

Since there are only 50 lines per submatrix, it should also be possible to return to a shared discriminator solution without loosing to much performance.

Preferentially, all of the zero suppression circuitry should fit into one tier, as shown in Figure 5.20, but due to the sliced nature of this circuitry, it should also be possible to have one tier for SDS and multiplexer and another tier for memory management and data transmission.

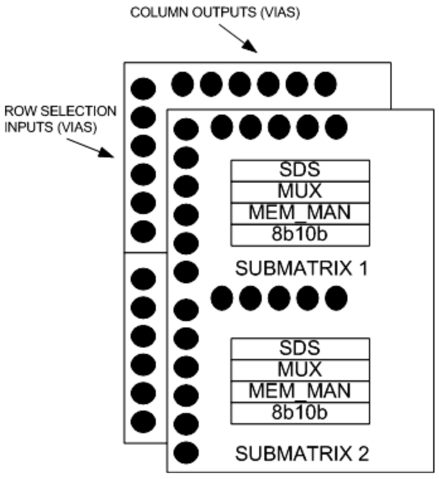


Figure 5.20: A folded SUZE with the zero suppression circuitry on top of the sensor, avoiding the dead area in the periphery.

If SUZE is to operate with 40 ns per line, then the system clock must increase its frequency from 100 MHz to 400 MHz! Such an increase in frequency should in theory be possible [6], but it might be a design challenge to meet timing requirements. However, the architectural modifications required for a folded SUZE on top of a rolling shutter tier is to be presented and discussed in the following:

5.3.1 Dimensioning SDS and multiplexer

To obtain the required dimensions, and to provide a basis for comparison between the two solutions, we start by providing the same hit model for the folded SUZE as we did for the first architecture. An average hit density of 7.2 $hits/mm^2/frame$ is therefore assumed throughout a the line.

Assuming pixel pitch of 18.45 µm and a line segment width of 64 pixels, the line segment size in mm^2 becomes:
$((18.45 \cdot 10^{-6}) \cdot (18.45 \cdot 10^{-6}) \cdot 64)/10^{-6} = 0.0218\ mm^2$.

For 64 pixels, the average number of hits in the worst-case hotspot becomes:

$hits\_per\_square\_millimeter\_per\_frame \cdot square\_millimeters\_per\_block$
$= 7.2 \cdot 0.0218 = 0.157\ hits.$

Still assuming that two states are used to store one hit, and thus being able to extract 3 hits from each segment, the probability of missing hits in one segment becomes $2.24 \cdot 10^-5$. Having sixteen such segments of worst-case hit density, the probability of missing hits in the entire line becomes $3.6 \cdot 10^{-4}$. The existing SDS structure of 6 states (3 hits if two states per hit are assumed) is therefore sufficient to ensure a hit miss probability less than 10^{-3}.

The same calculations have to be made for the multiplexer that selects states for the SDS circuitry.

With 1024 pixels in each line, the line surface becomes:

$((18.45\,\mu\text{m}) \cdot (18.45\,\mu\text{m}) \cdot 1024) = 0.3486\ mm^2$.

Still assuming the worst-case hotspot density throughout the entire line, the average number of hits per line becomes:

$$7.2\ hits/mm^2 \cdot 0.3486\ mm^2/line = 2.56\ hits/line.$$

[6]Static timing analysis performed in Synopsys for SDS, MUX and Memory Management shows that the timing requirements are met with a slack of approximately 1 ns. The analysis was performed with typical operating conditions after synthesizing the behavioral MIMOSA26 models to Chartered Artisan standard cells.

With an average of 2.56 hits/line, it is necessary to take into account nine hits to obtain a probability of missing hits that is less than 10^{-3}. 18 states must therefore be be selected by the multiplexer.

With such a large multiplexer implied, it is tempting to try another approach and consider the actual hit densities of a pixel line. By analyzing the simulation results shown in 5.2 and comparing with the proposed chip orientations of Figure 5.1, one finds that the worst-case arrangement of a sensor chip gives a worst-case average density of 3.6 $hits/mm^2$ inside of one line. The average number of hits per line then becomes:

$$3.6\ hits/mm^2 \cdot 0.3486\ mm^2/line = 1.28\ hits/line.$$

With an average of 1.28 hits/line, it is necessary to take into account six hits to obtain a probability of missing hits that is less than 10^{-3}. 12 states must therefore be be selected by the multiplexer.

Conclusion is therefore that while the SDS circuitry can be used as it is, the multiplexer structure has to be redimensioned to take into account 12 states instead of only 9.

5.3.2 Dimensioning the memory

For the memory, up to 12 states and 1 line address are to be stored during one line. With the memory management scheme of MIMOSA26/Ultimate, the write operations are synchronous to the CKREADPIX clock, which during one line is asserted six times. The first CKREADPIX cycle is used to provide valid addresses for the data that is to be written in the next cycle, and there are therefore enough clock pulses to write to the two memories five times, and thus store 10 words into memory.

To store 13 words in 2 parallell memories, it is necessary to provide 7 clock cycles. The CKREADPIX clock and the memory interface signals that are sensitive to this clock must therefore be replaced with another memory interface that is capable of performing 7 write cycles during the 40 ns of line processing, and whose frequency would be minimally 180 MHz.

Alternatively, one may implement three memories instead of two, and continue using the current 6-cycle CKREADPIX clock to write up to 6x3 words to memory.

5.3.3 Floorplanning and layout (area estimations)

The height of a submatrix with 50 lines becomes 18 µm · 50 = 900 µm.

Preliminary layout results, with the SUZE architecture implemented in Chartered process and with Artisan standard cells, shows a total height of SDS, multiplexer and memory management of 530 µm. A 512 words deep dualport-memory has a footprint with heigh of 156.4 um and width of 799.5 um. Total height with MUX, SDS and memories with their interfaces is therefore 687 µm.

It is thus sufficient area to use one tier for folding SUZE on top of a segmented rolling shutter chip.

5.3.4 Serial output links

The same calculations apply for the folded SUZE as for the first architecture, except that the option of direct transmission is not possible for the SUZE based architecture. To read out a ping-pong memory during 2 µs, five readout links are necessary for each submatrix.

5.3.5 Power consumption

Logic

Rough estimations on power consumption were obtained from the Encounter Power Analysis tool. The clock frequency was set to (6 CKREADPIX cycles / 40 ns) = 150 MHz. For the SDS, one assumes a switching rate of 0.1 for the SDS (most inputs do not contain hits), while for the MUX and memory management, a default switching rate of 0.2 is assumed:

SDS: $64 \cdot 0.30\,mW$ = 19.2 mW for 64 SDS blocks.
Multiplexer: 12.0 mW
Memory management: 11.1 mW

Total power consumption for one submatrix: 42.3 mW.

Total power consumption for 10 submatrices: 0.42 W

Memory

Two dualport-memories of size 512, with 7 write cycles during 40 ns, give an average frequency of 175 MHz. For reading of the memories, a frequency of 100 MHz is assumed:

Write AC current (1.5 V supply): $11.153\,mA$ => 16.73 mW.
Read AC current (1.5 V supply): $4.170\,mA$ => 6.255 mW.

Total dynamic power of two submatrix memories: 2 x (16.73 + 6.255) mW = 45.97 mW.

Total dynamic power of memories in 10 submatrices: 10 x 45.97 mW = 0.56 W.

More than half of the total power consumption is due to the dynamic power of the memories!

5.3.6 Scalability of architecture

As a starting point, we assume the token propagation path through the NAND/NOR sparse scan chain to be the critical path in the design. To minimize the delay through the NAND/NOR chain, the largest size NAND/NOR gates are chosen (X8), with the purpose of minimizing delay due to wire capacitance and other gates connected to the NAND/NOR outputs. The average delay through a pair of NAND/NOR-cells is therefore assumed to be the intrinsic and load-dependent falling delays of NAND and the intrinsic and load-dependent rising delays of NOR, with the two gates seeing the 0.016 pF loads of each others.

Based on the numbers obtained from the datasheet, one then arrives at an average delay of 0.047 ns/cell, and the time to pass through the 64 pixels of one sparse scan bank therefore becomes $64 \cdot 0.047 \approx 3\ ns$.

With the line processing time of MIMOSA26 reduced from 160 ns to 40 ns, the corresponding CKREADPIX clock period and the critical time between CKREADPIX and RSTPIX is reduced from 20 and 10 ns to 5 and 2.5 ns. Already with rough hand calculations, one sees that the sparse scan of SUZE could be timing marginal with the reduced line processing time.

However, by replacing the original sparse scan cells with improved sparse scan cells, proposed in Chapter 3, the critical time is increased from 2.5 ns to 5 ns, giving sufficient margins for the token propagation in the 64-bit segment.

For the following scalability calculations, the improved sparse scan chain is assumed.

Reducing segment size and increasing number of extracted states per segment

To obtain information on the scalability of the architecture, we make the same assumption as for the first architecture and expect the same average hit density in all the segments. To have a probability of missing hits in the entire line that is below 10^{-3}, the probability of missing hits in the individual segments must be below $6.5 \cdot 10^{-5}$. With each segment being able to handle 3 hits, this corresponds to an average of 0.20 hits/segment, which in the next turn translates into 9.18 $hits/mm^2/frame$.

With such a hit density over the entire line, the average number of hits per line would become, 3.33, and 10 hits per line would have to be taken into account. With 2 states per hit, a multiplexer that selected up to 20 states per line would have to be constructed.

Similar calculations have been done for the maximally achievable hit densities when the segment sizes are reduced to 32 and 16 pixels, and with the number of extracted states per segment increased to the limit given by the token propagation delay.

Table 5.2: The hit densities that can be covered with the SUZE based approach are shown as function of line segments and line buffers.

Line segments	**16**	**32**	**64**
Line buffers	6	12	30
Miss probability/segment (Maximally allowed)	$6.5 \cdot 10^{-5}$	$3.2 \cdot 10^{-5}$	$1.6 \cdot 10^{-5}$
Average of hits/segment.	0.20	0.85	4.4
Average of hits/mm^2. (Maximally allowed)	9.18	78	808
CKREADPIX frequency	200 MHz	400 MHz	1 GHz
States seen by MUX	96	384	1920
States to select by MUX	20	90	668

As seen from the table, the 16-segment SUZE solution is maximally capable of covering a hit density (9.18 $hits/mm^2/frame$) that is one third of the maximal density (28.12 $hits/mm^2/frame$) of the four-segment line buffer solution, and the 32-segment SUZE solution is similarly well below the hit density performance of the 8-segment line buffer solution. Only after increasing the number of segments to 64, the hit density performance of the SUZE sparse scan becomes better than the correspondingly tabulated performances of the line buffer solution.

Interfacing the increasing number of state outputs

Although an increased number of segments increases the covered hit density in the sparse scan chain, it should from the table also be clear that increasing the number of segments in practice only moves the problem from one stage in the data path to the next one. An increased number of segments implies increased multiplexer structures with increased amounts of states selected at the same clock cycle, and at a clock frequency approaching the gigahertz range. The high density scenarios with segments smaller than 64 bits are therefore not practically realizable, and one can conclude that the SUZE architecture has reached the end of scaling with respect to bank size and states to extract from each bank.

5.4 Conclusion with comparison of solutions

The most significant advantage of the token-based architecture is the improved timing margins given by parallellization:

- With the sequential sparse scan approach of SUZE, there is 40 ns available to process each line. To meet the line processing requirements, each line splits into minimally 16 banks. The large number of banks impose a large multiplexer to collect all the state ouputs of the banks, and this makes it difficult to meet the timing requirements.
- With the parallell token based approach, the token has 1 µs to fly through a line segment, and 4 segments per line is therefore sufficient to meet the timing requirements. It is thus possible to increase the number of segments per line to scale with a higher density of hits.

The pipelined nature of the readout also avoids the limitations given by the state selecting multiplexer of SUZE:

- The first architecture cannot take advantage of the hit densities reached by the sparse scan circuitry without having a similary large multiplexer. To comply with the CBM requirements, the maximum number of selected states per line must be increased to 12, which imposes a larger multiplexer and increasing difficulties to meet the timing requirements (and to arrive at a multiplexer size of 12, instead of an even larger one of size 18, one had to assume a specific chip orientation, which is dangerous as the chip orientations are preliminary and not final).
- With the pipelined readout of line buffers, there is no fixed limitation on how many states to read from a line, only an upper limit on how many states to read during the entire half-frame (dependent the number of states extracted per clock period and the FSM clock frequency). The last architecture can therefore be optimally balanced with a number of segments and buffers dimensioned for very high hit densities in only certain parts of the chip, and with the readout FSM dimensioned for a more relaxed average hit density in the entire matrix.

Finally, the power consumption results are also superior, compared to SUZE:

- While the SUZE based implementation is estimated to consume 1 W, the similar estimations for the token-based architecture with line buffers arrive below 0.3 W.

Although both proposed architectures meet the requirements that have been set for the CBM MVD, it seems like the architecture based on SDS and multiplexer is reaching the end of scaling with respect to hit densities, while the parallel token based architecture is only at the beginning of scaling.

Other differences that should be mentioned are that:

- The parallel token based approach does not require memory for buffering, while this feature is strictly necessary for the SUZE based implementation.
- In the case of a buffered solution with memory, both designs will require the same amount of serial readout links.
- Both designs will require the same amount of 3D stacked tiers as long as a separate multiplexing tier is required to give SUZE access to the pixels of the different lines. On the other hand, if the sensor tier can provide the lines directly to the next tier, the solution of a folded SUZE will require one tier less than the first solution.

Chapter 6

Summary and outlook

The starting point for this work was to study possible techniques that could make it realizable to come up with an architecture meeting the requirements for the most challenging physics experiments such as CBM. Legacy pixel detector architectures have been a compromise between spatial resolution, time resolution, and radiation hardness. Since CBM demands the best performance in all these fields, none of the existing architectures were capable of meeting the CBM requirements.

Starting in spring 2008, the state of the art within MAPS readout was column-parallell readout (rolling shutter, MIMOSA22), without any kind of data sparsification. My first contribution to the development of MAPS based readout architectures was therefore to embed row-wise zero suppression and a serial transmission interface into the next sensor that was under development, the MIMOSA26. MIMOSA26 provided a frame readout time of 100 µs for a matrix of approximately 500,000 pixels.

100 µs are still far above the few microseconds required for CBM MVD. The frame readout time can be reduced by reducing the number of rows. But as material budget constraints do not allow the overlap of more than two detectors, the minimum height of a pixel matrix equals the height of the readout electronics, and it has been shown in [39] that the highest expected time resolution of a minimum-height, rolling shutter operated two-dimensional MAPS circuit will be around 10 microseconds.

However, with 3D integration, this limitation no longer applies, as the readout electronics can be put on top of the matrix and the dead area can be removed completely. 3D integration also gives the possibility to combine a fast readout structure with a sensor of high-resistivity epitaxial, thus overcoming the previous short-stopper of massive radiation damage when standard low-resistivity MAPS are placed in the radiation environment of CBM.

With the first IPHC 3D detector prototypes, there has been two main approaches:

The first one, application specific for the ILC bunch train, has been to keep all pixels powered on during the 1 ms bunch train, time tag the hits using timestamp flip flops in every pixel and then perform a full readout of all pixels between the bunch trains.

The second one is still based on continuous readout with rolling shutter operation of the pixel matrix, but it takes advantage of 3D integration to embed a discriminator in every pixel, instead of the shared column-discriminator used in MIMOSA26. Embedding discriminator in each pixel reduces both power consumption and improves line processing speed (from 160 ns in MIMOSA26 to 50 ns in the first prototype).

Although taking part in the development of the first chip, and being responsible for its serial readout interfaces, it was for the continuation of this work necessary to pursuit the second approach, as its principal focus was the CBM with continous beam and therefore no opportunity to use timestamping with delayed readout.

The future prospect for the last approach is that the line processing time can be reduced even further, down to 40 ns, and that the desired frame readout time is obtained by further dividing the pixel matrix into submatrices that are each rolling shutter operated in parallel. What was left to explore, seen from a digital design point of view, was what kind of digital readout architecture to put on top of this structure.

Already being familiar with the token based sparse scan readout of MIMOSA26, an improved implementation of the sparse scan that increases speed and minimizes logic was proposed. Compared to the sparse scan of MIMOSA26, the proposed new sparse scan was in fact sufficiently compact to fit into a structure of digital small-pitch pixel cells, and by distributing the search for hit patterns into the individual pixels, we parallelized the tasks of zero suppression, previously performed sequentially.

To combine parallel search for hit patterns with sequential processing of the lines of the submatrices, each submatrix splits into two halves, with rolling shutter injection of hits in one half and token-based sparse scan for hit patterns in the other half. With this approach, the critical time per line increases from the 160 ns in MIMOSA26 to 1 µs in the new conceptual design. Due to the relaxed time for the tokens to pass through the sparse scan chain of a row, architectures based on such an approach does not only have a much smaller frame readout time than MIMOSA26, they can also be redesigned to scale with a much higher hit density.

Utilizing the new and emerging techniques and technologies, like 3D integration to combine fast and parallellized digital readout with high-resistivity, radiation tolerant MAPS, and fast 8b10b transmission to transfer large amounts of data with a minimum of bonding wires, a concept for a 3D integrated detector targeted against CBM has thus been developed.

To build such a 3D integrated detector, there will still be years of prototyping for the sensor and analog part of the design, but a concept for the readout has been developed and verified through simulation.

Before interfacing the digital readout to an actual sensor tier, the next step of functionally validating the architecture could be to implement the readout in a single digital wafer, with the readout buffers placed in the periphery of the matrix, and with a scan chain of flip flops, implemented custom or as JTAG, to emulate the discriminated hit signals. New patterns can be shifted into the flip flops between each frame read out. Such a test design, with solely digital components and with hit patterns inserted through a scan chain, could as well be implemented in an FPGA.

The different building blocks of the circuit should be implemented and tested in the years to come, so the readout electronics is ready when the other parts of the detector are developed and mature for a first prototype or a final implementation.

Appendix A

Publications

This appendix contains the list of publications I have authored and co-authored.

O. Torheim, K. Bronstad, K. Heerlein, U. Mall, A. Nathues, W. Nowosielski, P. Orleanski, B. Pommeresche, V. Reimundo, Y. Skogseide, A. Solberg and K. Ullaland, *"Development of an embedded CPU-based Instrument Control Unit for the SIR-2 Instrument on board the Chandrayaan-1 Mission to the Moon"*. IEEE Transactions on Geoscience and Remote Sensing, Volume 47, Issue 8, pages 2836-2846.

Ch. Hu-Guo, J. Baudot, G.Bertolone, A. Besson, A.S. Brogna, C.Colledani, G. Claus, R. DeMasi, Y. Degerli, A. Dorokhov, G. Doziere, W. Dulinski, X. Fang, M. Gelin, M. Goffe, F. Guilloux, A. Himmi, K. Jaaskelainen, M. Koziel, F. Morel, F. Orsini, M. Specht, Q. Sun, O. Torheim, I. Valin and M. Winter. *"First reticule size MAPS with digital output and integrated zero suppression for the EUDET-JRA1 beam telescope"*. Nuclear Instruments and Methods in Physics, doi:10.1016/j.nima.2010.03.043.

A. Himmi, G. Doziere, O. Torheim, Ch. Hu-Guo, M. Winter. *"A Zero Suppression Micro-Circuit for Binary Readout CMOS Monolithic Sensors"*. Proceedings at TWEPP-09.

Y. Fu, O. Torheim, Ch. Hu-Guo, M. Winter, *"3D Integrated Monolithic Active Pixel Sensors for Charged Particle Detection"*. To be submitted to Journal of Instrumentation.

Y. Fu, O. Torheim, Ch. Hu-Guo, Y. Hu, M. Winter, *"A vertically integrated 3D CMOS monolithic active pixel sensors with fast digital pipelined readout"*. Presentation at the VIPS2010 Workshop, 22nd to 24th of April in Pavia, 2010.

O. Torheim, *"3D-integrated MAPS for the CBM-MVD"*. Presentation at 15th CBM Collaboration Meeting, 12th to 16th April in Darmstadt, 2010.

O. Torheim, Y. Fu, Ch. Hu-Guo, Y. Hu, M. Winter, *"3D CMOS monolithic 3-bit resolution pixel sensor with fast digital pipelined readout"*. Presentation at the Journees IN2P3 VLSI-PCB-FPGA, 21st to 24th of June in Paris, 2010.

Y. Fu, O. Torheim, Ch. Hu-Guo, Y. Hu. *"A CMOS monolithic 3-bit resolution pixel sensor based on 3DIT for the innermost layer of the ILC vertex detector"*. To be submitted to Journal of Instrumentation.

O. Torheim, *"3D-integrert elektronikk. Nye dimensjonar, nye perspektiv"*.
Popular scientific article in Elektronikk, Issue 3, 2010.

Appendix B

Comparison of clusterization and state encoding with JPEG

In this appendix, clusterization and state encoding is qualitatively compared with JPEG.

With JPEG compression, an image bitmap is taken through the following steps (see for example [95]):

- Discrete Cosine Transform (DCT) and suppression of high frequency spatial components.
- Run Length Encoding (RLE)
- Huffman encoding

With the first stage, DCT, the image is transformed into a spatial frequency domain. Then the harmonics are either totally suppressed or stored with a reduced resolution. This first stage of JPEG compression is lossy, but it can be done without noticeable degradation of image quality due to the sensibility of the human eye.

DCT and frequency suppression is therefore ruled out if the entire information is to be preserved, as may be necessary for an particle trajectory sensor in a high energy physical experiment. If any kind of information was to be filtered out, it had to be noisy pixels, and for these tasks an algorithm analyzing identified clusters would be a better approach. But what about RLE and Huffman encoding?

The principle of RLE is to code series of symbols. For example, if we have a series of letters, like AAAAAAAAAABCDEEEEEEEEEE, which is a series of 10 A, 1B, 1C, 1 D and 10 E, we can RLE encode this as 10A01B01C10D10E. In other words, RLE may reduce the length of a symbol series, as long as the same symbols repeat themselves often enough. But the opposite may also be the case, if we have a series with few or none repetititions like ABCDEFG, then our RLE code - 1A1B1C1D1E1F1G - would have been longer than the uncoded series. However, in a binary discriminated pixel matrix, one would expect to find a few ones in an ocean of zeroes (hits due to particles and fake hits due to noise should amount to a few percent of the pixels with moderate occupancy), and run length encoding would therefore be an efficient way to compress the pixel matrix.

The next stage is Huffman encoding. This encoding is efficient if some symbols appear more frequent than others. With Huffman encoding, we build a binary Huffman tree bottom up, where we systematically construct new nodes by connecting the least frequent nodes to new nodes, until all the nodes are connected together at a top level. Then we assign bit values to all the different nodes, and the nodes with most branches downwards in the hierarchy (the least frequent nodes) are encoded with most bits. The most frequent symbols are thus encoded with more bits than the lesss frequent ones.

The backdraft of both RLE and Huffman encoding is that this kind of encoding is optimal when the entire image can be analysed in total. For example, how many bits should be used as coding bits and how many bits should be used for storing the symbols in the RLE encoding? The answer to this question depends on how often data is changing, and the entire image must therefore be analysed in advance. The same is the case with Huffman encoding, the Huffman tree must be built on the basis of the entire image if the Huffman encoding should be optimal. However, if the shape of the images is not totally random, it is also possible to use experimental data from previous images as a starting point for constructing an RLE encoding scheme and a Huffman tree. With this approach, it is also possible to imagine an image being RLE and Huffman encoded directly, row by row.

It is easy to see that for the case of a pixel detector, where most of the matrix is left blank, RLE encoding would in practice be the same as zero suppression. If a row containing one state had been RLE encoded, it would start with a counting of zeroes, then the counting of zeroes and ones inside a state and finally counting of the rest of the zeroes in the row.

If we compare run length encoding to state encoding, we see that the address field in an encoding state could be compared to the corresponding counting of previous zeroes in an RLE encoding scheme. However, with state encoding, we base our encoding on a physical model where we know that there are much fewer ones than zeroes, and we do not expect more than four in a series. We can also make the assumption that a line is most likely to start with a series of zeroes and not with ones. Since the matrix is encoded row by row, with a fixed length of each row, it is also not necessary to count the number of zeroes following the last series of ones.

Since we alway assume that our line will start with a long series of zeroes, followed by a shorter series of ones, with this cycles repeating until the last short series of of ones, it is possible to modify the RLE scheme in the following way:

- Fewer bits may be used for length encoding of ones than for zeroes.
- Since there is always a cycle starting with zeroes and ending with ones, we can suppress the bits containing the symbols, and only include then counting bits. A situation where we should start with ones instead of zeroes can be modeled by coding zero zeroes before starting to encode the series of ones.
- For the counting of zeroes, we therefore need 11 bits for counting from zero to 1024 (worst case with empty line). For the coding of ones, where the number of ones is always within the range of one to four, we only use two bits, with 00 representing one and 11 representing four.

With this approach, our modified RLE encoding starts to remind very much of zero-suppressed state encoding. An example is shown in the table below:

RLE	**00010100000**	**11**	**00000001000**	**10**
	(no. zeroes)	(no. ones)	(no. zeroes)	(no. ones)
Zero sup.	**00010100000**	**11**	**00010101100**	**10**
	(state addr.)	(state enc.)	(state addr.)	(state enc.)

From this comparison between zero suppression and a modified RLE, we see that the number of zeroes in RLE in practice is the same as the addressing of states in zero suppressed state encoding. The difference is that with zero suppression, the states are addressed absolutely, while the modified RLE addresses the states relative to the previous state. The conclusion must therefore be that zero-suppressed state encoding may be viewed as some kind of improved RLE encoding, where the physical background with series of small clusters in an ocean of zeroes is taken into consideration.

When clusterisation is used instead of state encoding, the result gets even better, since we do not have to store almost simular colums addresses three times (one addressed cluster instead of three addressed states with more or less the same column address).

So what about the final stage with Huffman-encoding? Would there be something more to win by implementing this technique as a last compression stage following clusterisation? As already mentioned, techniques like Huffman encoding are based on coding the most frequent symbols with fewer bits than the rare ones.

If we consider an encoded cluster, this cluster is split into its address field and its cluster encoding field. Regarding the address bits, we consider it unlikely that some kinds of column address should appear more frequently than other addresses

- since our physical model predicts a uniform distribution of hits across the matrix. But what about the encoded cluster? Could there be any kind of clusters that are more frequent than other and more rare clusters?

For example, if experimental results had shown that regular clusters represented 90 % of the hits, an easy technique to reduce the amount of data could be to use only one bit for encoding regular cluster, while at the same time increasing the number of bits for storing rare clusters from 8 to 9. Such an approach could in the best case reduce the size of addressed and encoded clusters from 30 bits (11 + 11 + 8) to 23 bits (11 + 11 + 1), which is an improvement of 23 % (in the most optimistic case where we always have regular clusters and never anything else).

However, such an approach would result in different lengths on the addressed and encoded clusters. Instead of a simple memory interface that stored each addressed and encoded cluster into each the memory element, there would be need for additional logic to collect clusters of different length into one long word that was to be stored in several memory elements in an overlapping fashion. A look-up table to contain the Huffman tree for the 256 combinations of 8-bit words, must also be implemented. Such a memory could require considerable size, dependent on the shape of the tree and its implementation (See for example [96] or [97]).

Appendix C

Control registers for MIMOSA26

This appendix contains an overview of the MIMOSA26 control registers.

The JTAG registers that are used to control the behavior of the chip are the following:

- SEQ_PIX_REG(127:0). Contains registers DCALIB, DREAD, DRST, DCLAMP, DLATCH and other 16-bit registers that control the timing of control signals for the analog data path.
- SEQ_SUZE_REG(159:0). Contains registers DCKREADPIX, DCKMEMLATCH, DRSTPIX and other 16-bits registers that control the timing of control signals for the digital data path. The sequencer register is the longest register of them all, as it can be split into a variety of smaller registers that are used inside the sequencer for controlling the readout sequence.
- RO_MODE0(7:0). This register is 8 bits wide, and each of the bits is used to select specific digital modes of operation for the chip. For example, the test mode where the SS/SE alternates its inputs between the two pixel matrix line registers instead of processing the real pixel matrix. Or another test mode where the chip is set to process only half of the matrix, 320 rows, instead of the 576-row matrix. A third example could be the disabling of the LVDS outputs.
- RO_MODE1(7:0). This register is 8 bits wide, and each of the bits is used to select specific analog modes of operation for the chip. For example, bit 0 is used to enable the test mode for the discriminators.
- DIS_DISCRI(1151:0). This register is used to disable individual discriminators in a line. Such disabling may be necessary if a large amount of the pixels in the corresponding column are noisy.

- BIAS_GEN(151:0). Contains registers to control the 16 DAC units of the analog circuitry, where each DAC is of 8 bits resolution. For example control of the DACs that give out the VREF1 and VREF2 voltages.
- PIXEL_MATRIX_LINE0(1151:0) and PIXEL_MATRIX_LINE1(1151:0). These registers can be loaded for test purposes and the digital data path can be tested with known input content when the circuit is run in a test mode where the input to the SS/SE circuits are alternated between the two line registers. The two pixel line registers contains test pixel data for two lines. These registers are used to inject test pixel data that can be processed by the chip. When the chip is running in test mode, the two lines in the test registers are processed consecutively, one line at a time, over and over again.
- CONTROL_PIX_REG(39:0). Contains different test parameters like control of which signals that are to be routed out on test pads.
- HEADER_TRAILER_REG(63:0). The header and trailer registers contain two 16-bit header registers and two 16-bit trailer registers. The two pairs of headers and trailers are used as headers and trailers for each their respective LVDS link.
- CONTROL_SUZE_REG(47:0). This register contains control registers for different functions. First, there are registers to control which internal memory handling signals that are to be routed out of the chip for test purposes. Then, there are registers to command the chip to sample the content of specific stages in the digital datapath and transfer this data through the LVDS link instead of the normal data. Finally, there are registers to control the LVDS modes of transfer.

Appendix D

Test patterns for memory and datapath of MIMOSA26

This appendix contains test patterns used for testing memory and datapath.

D.1 Memory test structures

An optional memory test can be performed in the seven first frames of chip operation. A fixed set of patterns are written into the pingpong memories on the first frame and read back on the next frame. The data words read back from memory are XOR'ed together with the originally written words and the bits in the result strings are OR'ed together and routed out to a test pad.

The different stages in the memory test are detailed below:

Frame 1:

Write zeroes to memory 0 and 1.

Frame 2:

Write zeroes to memory 2 and 3.

Read data from memory 0 and 1 and XOR it with zeroes.

Frame 3:

Write 5A5A to memory 0 and 1.

Read data from memory 2 and 3 and XOR it with zeroes.

Frame 4:

Write 5A5A to memory 2 and 3.

Read data from memory 0 and 1 and XOR it with 5A5A.

Frame 5:

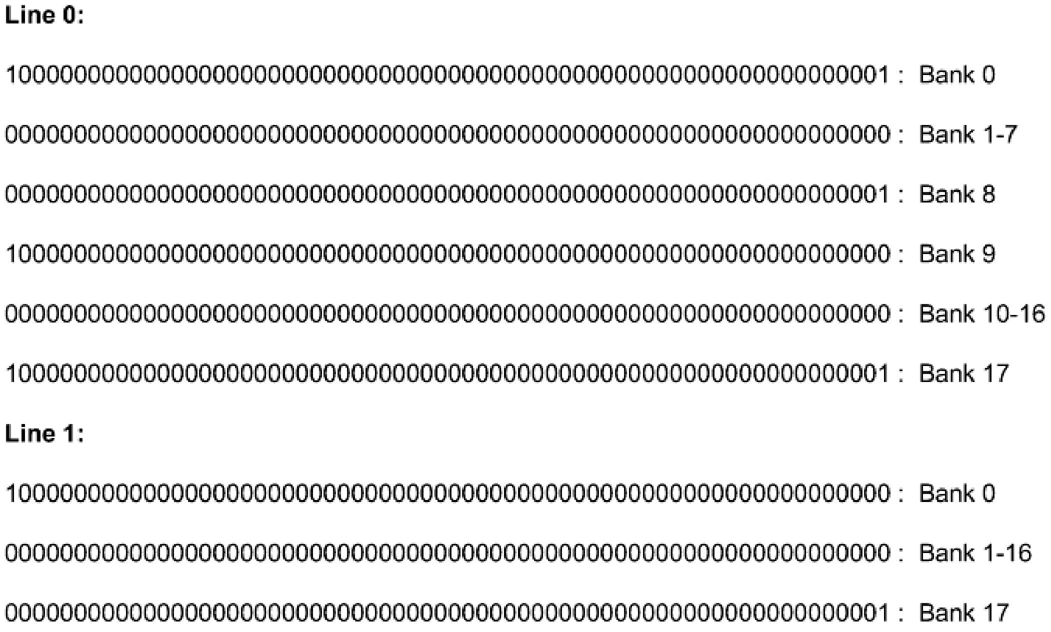
Write A5A5 to memory 0 and 1.

Read data from memory 2 and 3 and XOR it with 5A5A.

Frame 6:
Write A5A5 to memory 2 and 3.
Read data from memory 0 and 1 and XOR it with A5A5.
Frame 7:
Read data from memory 2 and 3 and XOR it with A5A5.
With the selected set of test pattern, every bit in the memory has been toggled.

D.2 Two-line patterns for propagating failure in the datapath

Figures D.1, D.2, and D.3, are two-line patterns designed to activate a wide range of possible failures in the datapath.



Line 0:

100000000000000000000000000000000000000000000000000000000000000000000001 : Bank 0

000000000000000000000000000000000000000000000000000000000000000000000000 : Bank 1-7

000000000000000000000000000000000000000000000000000000000000000000000001 : Bank 8

100000000000000000000000000000000000000000000000000000000000000000000000 : Bank 9

000000000000000000000000000000000000000000000000000000000000000000000000 : Bank 10-16

100000000000000000000000000000000000000000000000000000000000000000000001 : Bank 17

Line 1:

100000000000000000000000000000000000000000000000000000000000000000000000 : Bank 0

000000000000000000000000000000000000000000000000000000000000000000000000 : Bank 1-16

000000000000000000000000000000000000000000000000000000000000000000000001 : Bank 17

Figure D.1: Two-line test pattern for critical path activation.

Line 0:

```
010010101110100010001110111111111110000000011111111111111111111 :  Bank 0
100001110001100001011011101111111110000000011111111111111111111 :  Bank 1
000000110000110101101100001111111110000000011111111111111111111 :  Bank 2
000000000000000000000000000000000000000000000000000000000000000 :  Bank 3-17
```

Line 1:

```
001101000010101110001110011111111100000000111111111111111111111 :  Bank 0
011110110110001101010100010111111110000000011111111111111111111 :  Bank 1
000100110101100111111000111111111110000000011111111111111111111 :  Bank 2
000000000000000000000000000000000000000000000000000000000000000 :  Bank 3-17
```

Figure D.2: Two-line test pattern for testing left half of matrix.

Line 0:

0000000000000000000000000000000000000000000000000000000000000000 : Bank 0-8

0001100010000010000100101101111111000000001111111111111111111111 : Bank 9

0001011101100000111001000111111111000000001111111111111111111111 : Bank 10

0000111001010011010101110011111111000000001111111111111111111111 : Bank 11

0000000000000000000000000000000000000000000000000000000000000000 : Bank 12-17

Line 1:

0000000000000000000000000000000000000000000000000000000000000000 : Bank 0-8

0011101111011000110011011110111111000000001111111111111111111111 : Bank 9

0001011100011100000100100001111111000000001111111111111111111111 : Bank 10

0011100110001000101001110111111111000000001111111111111111111111 : Bank 11

0000000000000000000000000000000000000000000000000000000000000000 : Bank 12-17

Figure D.3: Two-line test pattern for testing right half of matrix.

Appendix E

ILC perspectives for MIMOSA26

This appendix presents the perspectives of adapting the MIMOSA26 architecture to the ILC inner and outer layers.

E.1 ILC: Relaxed data transmission between bunch trains

It is predicted that the MIMOSIS-1, when implemented in new and smaller technology, may reach a frame readout rate of 25 µs, and thus meet the frame readout rate requirements of ILC.

However, to make the MIMOSA26 line of sensor architectures comply with the requirements for ILC, it is necessary to adapt the state selection logic to a higher density of hits per line, and to modify the buffering and data transmission scheme to take advantage of the periodicity of the ILC bunch train. We have explored which changes are necessary and propose the following architectural adaptations:

E.1.1 Redimensioning the multiplexer or replacing state encoding with clusterization

As was already shown in Chapter 2, the number of hits per line that have to be taken into account for a sensor in the inner layer for the ILC vertex detector, amounts to 6. Still assuming a multiplicity corresponding to three states per hit, the total number of states per line then becomes 18, and thus twice as much as the number of states per line that the multiplexer of MIMOSA26 is dimensioned for.

To make the MIMOSA26 series of pixel readout architectures comply with the ILC requirements, it is therefore necessary to redimension the multiplexer, from selecting 9 states per line to selecting 18 states per line. Alternatively, if state encoding is replaced with clusterization, one may relax the selection from nine states to six encoded clusters - or replace nine states with nine encoded clusters to scale with even larger hit density.

E.1.2 Replacing continuous data transmission with delayed data transmission

With the ping pong memory structure of MIMOSA26/Ultimate, matrix processing and data transfer out of the chip is performed at the same time - with the previous picture being transferred from the first buffer while the current picture is being read and stored into the second buffer, and with a data rate corresponding to the worst-case amount of hits during one frame.

If every frame is to be transferred out of the chip in a 25 µs time interval, a transfer rate into the gigabit range is necessary, and it will represent a considerable synchronisation problem for the data acquisition system in the receiving end if it is to synchronise data at such rates from hundreds of sensors in the same time.

A better approach would be to take advantage of the duty cycle of the ILC bunch train, whose length is 1 ms and period is 200 ms. If all the frames aquired during a pulse train could be buffered on board the sensor chip, the data acquisition system would have all the 199 ms until the next train arrives to receive data.

For a time resolution of 25 µs, it is necessary to acquire 40 frames during 1 µs, and to make the MIMOSA26 line of architectures compliant with the requirements of ILC, it would therefore be necessary to increase the memories to a size large enough to store 40 frames. As readout and acquisition happens at each their time, it would no longer be necessary with a ping-pong configuration, and a single port memory of appropriate depth should instead be implemented.

For the inner layer of ILC, the average number of hits/frame/cm^2 is 300. With the area of one inner layer sensor being 1 cm^2, and the number of frames per bunch train being 40, the average number of hits per bunch train becomes 12,000. For a probability of missing hits that is less than 10^{-3}, 12,340 hits have to be taken into account.

Assuming 3 states for each hit, the number of states to store during 40 frames becomes $12,340 \cdot 3 =$ 37,020.

As each sensor is to be equipped with 256 rows, there will also be 256 row addresses to store, when minimum one hit in every line is assumed, and for 40 frames, the total number of rows due to line addresses becomes $256 \cdot 40 =$ 10,240.

As one word is used to store one state, and one word is used to store one line address, the total number of words to store during one bunch train becomes 10,240 words + 37,020 words = 47,260 words.

If clusterization is implemented instead of state encoding, there will be two words for every encoded cluster, and the total number of words to store due to the encoded clusters becomes $12,340 \cdot 2 =$ 24,680 (for clusterization, there are sufficient free bits to embed the line address into the 32-bit word used for column address and cluster code, and we therefore do not need to add 10,240 words of memory space for line addressing).

Due to material budget, the height of the memories should not considerably exceed the height of the memories used in MIMOSA26, which are of 600 µm. Using the memory compiler of Artisan, memories for Chartered-0.13 have been generated with sizes of 8192, 4096 and 2048. The area of the respective memories, and the number of memories required to store the 47,260 words for state encoding and the 24,680 words for clusterization, respectively, are tabulated in Table E.1.

Table E.1: The different memory sizes and their quantities required for states and clusters.

Size (words)	**Height** (µm)	**Width** (µm)	**Memories** (states)	**Memories** (clusters)	**Tot. width** states (µm)	**Tot. width** clusters (µm)
8192	1000	433	6	4	2598	1732
4096	545	427	12	7	5124	2989
2048	311	424	24	13	10176	5512

To have the same height of the memory management as in previous lines of the MIMOSA26 architecture, the memory of depth 4096 words should be chosen. For state encoding, a total of 12 memories must then be placed in parallell in the layout, which when placed alongside will occupy a total area of 545 µm x 5124 µm. For comparison, it can be mentioned that the four single-port memories used to make the ping-pong configuration of MIMOSA26 occupy a total area of 6000 µm x 600 µm.

However, instead of continuing the MIMOSA26 series of architectures, a new architecture based on 3D integration, with timestamp flip-flops in every pixel and delayed readout of all the flip flops between the bunch trains, has been chosen as the candidate for the inner layer sensors. This architecture is further described in Chapter 4. With timestamped hit information stored in every pixel, it is no longer necessary to memorize each frame. Still, a considerable disadvantage is introduced with delayed readout as only one hit per pixel can be stored with correct time information (second hits are tagged with an overflow bit and eventual third and fourth hits are not registered at all).

E.2 ILC outer layers: Lower power and higher pixel accuracy

The question of using small pixels and binary discrimator or larger pixels and ADC is a question of optimalization with respect to power consumption. If an ADC is implemented instead of a discriminator, the same single-point resolution may thanks to charge sharing be obtained with a larger pixel pitch. Fewer ADCs than discriminators are therefore needed, and with an optimum combination of pixel pitch and ADC bits, it is thus possible to save power.

For the two outer layers of the ILC vertex detector, it is planned to replace the column-level discriminators of Ultimate-1 with an ADC of 4 or 5 bits. Such an ADC has already been developed, and is further described in [98].

It has been part of this study to find out which architectural changes are necessary to make the MIMOSA26 and Ultimate architectures comply with a 4-bit ADC, and the results of this study are to be presented in this chapter. Given the fixed memory sizes of Ultimate-1, we have explored different types of encoding with the purpose of finding an implementation that is optimal in terms of logical simplicity and quantity of data to store in memory.

E.2.1 Requirements derived from physics experiment

The features of Ultimate-1, which is targeted for the STAR experiment, are depicted below:

- 1088 rows x 1024 columns
- Line-wise state encoding with maximum 9 states per line
- Ping-pong memory of 2x2048 words (16 bits)

The question to be answered is which modifications must be done to Ultimate in order to make the rest of the readout chain, with hit encoding and memory storage, comply with a 4-bit ADC?

The expected number of hits due to particles are given by the physics of the ILC experiment, and modelled as a stochastic process with Poisson distribution.

The ILC vertex detector is built up as a barrel composed of five layers, with L0 being the inner layer and L4 being the outer layer. The L3 and L4 sensors of ILC have active chip surfaces of 4 cm^2.

The average number of hits/cm^2/frame is 12 hits for L3 and 6 hits for L4 [99], and the average number of hits /chip/frame is therefore 48 hits(12 x 4) for L3 and 24 hits (6x4) for L4.

For a given average number of hits, λ, the cumulative probability function of the Poisson distribution can be used to find the maximally expected number of hits:

$$P(N > N_{MAX}) = P - P(N \lesssim N_{MAX}) = 1 - \sum_{N=0}^{N=N_{MAX}} \frac{\lambda^N e^{-\lambda}}{N!}$$

For providing a safety margin, we want to find out how many hits per chip that must be taken into account if we want a probability of missing hits that is less than 10^{-3}. Using the Poisson distribution function under this condition, the number of hits due to particles is expanded to a value of 71 for L3 and 40 for L4.

The expected number of fake hits can be written as:

$$number\_of\_pixels_{per\_chip} = \frac{chip\ area}{area\ of\ one\ pixel}$$

Given the chip area of 4 cm^2 and the pixel pitch 33 μm, the number of pixels per chip becomes:

$$number\_of\_pixels_{per\_chip} = \frac{4x(10^{-2})^2}{(33x10^{-6})^2} \approx 367,310 \quad pixels$$

Assuming the same low fake error rate as for MIMOSA26[1], below 10^{-4}, the maximum number of fake hits becomes $367,310$ x $10^{-4} \approx 37$.

The total number of expected hits is therefore:

Particles: Max. 71 hits/chip.

Fake hits: Max. 37, corresponding to a fake error rate of 10^{-4}.

One should therefore expect a worst case of 71 hits due to particles, in addition to 37 fake hits [2].

Based on the presented requirements, we propose the following types of encoding:

E.2.2 Data sparsification with state encoding

The most straighforward approach would be to continue with state encoding, but modifying the data storage format for also storing the 4-bit quantized values.

[1]Tests of MIMOSA26, performed with S/N cut of 6, shows an average of 40 fake hits[54].

[2]Because of a high detection efficiency rate above 99 %, we neglect the probability of not detecting an event.

It is necessary for the state encoding that the analog signal is quantized to binary values. In Figure E.1, it is shown that the binary quantization can be performed either through a discriminator stage parallel to the ADC or in a digital circuitry following ADC (First subtract to check if $A - Ref > 0$. Then discriminate by the MSB of the secondary-complemented result).

Which approach to use is a matter of the most convenient analog/digital partitioning.

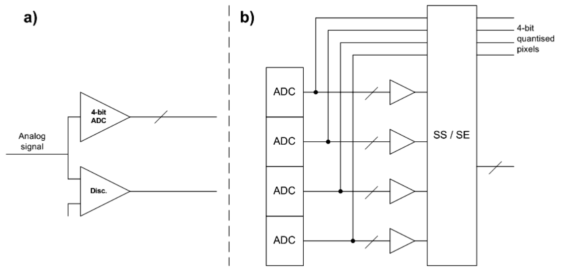


Figure E.1: a) Two separate branches in the analog circuitry. b) Binary discrimination performed in digital circuit following ADC.

A proposed data format, where the state encoding comes first, followed by the four A/D converted pixel values, is shown in Figure E.2.

State data format (2x16 bits)				
col. addr. (11 bits) + state enc. (2 bits) + 3 free bits	pixel 1 (4 bits)	pixel 2 (4 bits)	pixel 3 (4 bits)	pixel 4 (4 bits)

Figure E.2: Data format of encoded state with ADC data.

E.2.3 Data sparsification with clusterization

With a clusterization algorithm, the address of the central pixel is identified and one byte is used to identify up to 8 crown pixels around the central pixel. See Figure E.3.

It is important to note that while one encoded cluster contains information on 9 pixels, the three states identifying a similar cluster would contain information on 3x4=12 pixels!

Two different formats are shown in Figure E.4.

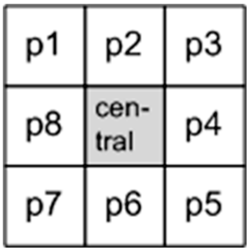
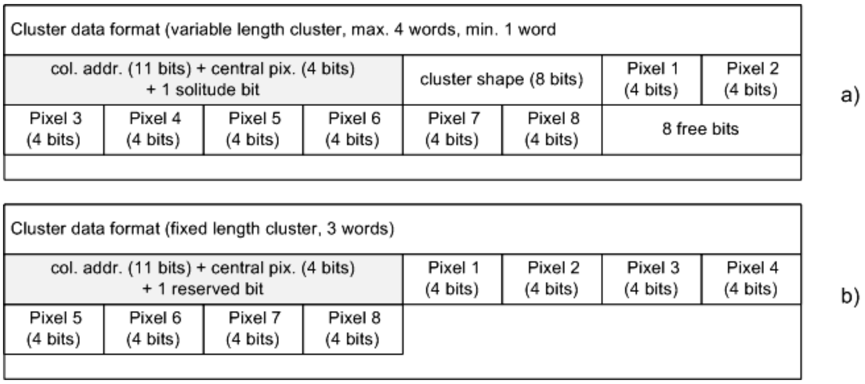


Figure E.3: Shape of an encoded cluster.



Cluster data format (variable length cluster, max. 4 words, min. 1 word							
col. addr. (11 bits) + central pix. (4 bits) + 1 solitude bit				cluster shape (8 bits)		Pixel 1 (4 bits)	Pixel 2 (4 bits)
Pixel 3 (4 bits)	Pixel 4 (4 bits)	Pixel 5 (4 bits)	Pixel 6 (4 bits)	Pixel 7 (4 bits)	Pixel 8 (4 bits)	8 free bits	

a)

Cluster data format (fixed length cluster, 3 words)							
col. addr. (11 bits) + central pix. (4 bits) + 1 reserved bit				Pixel 1 (4 bits)	Pixel 2 (4 bits)	Pixel 3 (4 bits)	Pixel 4 (4 bits)
Pixel 5 (4 bits)	Pixel 6 (4 bits)	Pixel 7 (4 bits)	Pixel 8 (4 bits)				

b)

Figure E.4: Data formatting of encoded clusters. a) Encoded cluster of variable length. b) Encoded cluster of fixed length.

E.2.4 Implementation and memory issues with state encoding

With the memory management of MIMOSA26, two 16-bit memories are written in parallel on 5 clock cycles during one line. Up to 10 words can therefore be stored into memory without changing the way the memory is interfaced.

If one continues to assume assume nine states per line for the state encoding, with two words for every encoded state, one arrives at:

9x2 words + 1 address line word = 19 words.

With two words being written into memory at each clock cycle, 10 clock cycles are therefore needed, twice as many as with MIMOSA26.

A new writing and clocking scheme for the memories must therefore be implemented, independent of the CKREADPIX clock that is used for the rest of the zero suppression circuitry. And with 200 ns line processing time, it is necessary with at least 100 MHz (10 ns) write operation!

Alternatively, if not considering the 2 x 16-bit memories as a given constraint, one can increase the width of each of the memories from 16-bit to 32-bit, and store each state as a 32-bit word. If two states (four words) stored at every clock cycle, it is still sufficient with the 5 clock cycles of the MIMOSA26 memory management scheme.

E.2.5 Implementation and memory issues with clusterization

Assuming nine states per line, one can also assume three clusters per line.

With an encoded cluster of variable length, there can be up to 4 words per encoded cluster. The maximum number of words to store into one memory during one line will therefore become:

3x4 words + line address = 13 words.

We therefore see that it is necessary to increase from 5 to 7 clock cycles during one line. Even if only a slight increase in the number of clock cycles is required, it makes it necessary to rewrite the memory interface, as the 6-cycle CKREADPIX can no longer be used for the memory. Alternatively, one can expand the memories to 32-bit to write the 13 words in 4 cycles.

Variable length also implies more complex logic than otherwise necessary for fixed length.

With an encodd cluster of fixed length, there are always three words to be written per cluster. The maximum number of words to store into memory during one line will therefore become:

3x3 words + line address = 10 words.

Current memory management for state encoding already runs 5 clock cycles on two parallel 16-bit memories during one line. With 10 words to store, and 2 words being written in parallell at each clock cycle, the fixed-length clusterisation therefore fits perfectly into the existing memory management scheme!

E.2.6 Memory storage considerations

Dependent on the type of encoding, there will be different requirements on memory storage. To store one particle, assuming a regular cluster[3], the number of 16-bit words per particle becomes:

State encoding with 3 states per hit: **6 words**
Fixed-length cluster: **3 words**
Variable-length cluster: **3 words**

To store one fake hit, the number of words becomes:

State encoding with 1 state per fake hit: **2 words**
Fixed-length cluster: **3 words**
Variable-length cluster: **1 word**

To store one particle, assuming a reduced cluster multiplicity of 3, the number of words per particle becomes:

One to three states, dependent of shape: **2-6 words**
Fixed-length cluster: **3 words**
Variable-length cluster: **2 words**

To explore the memory storage capabilities with 2x2048 words, one assumes the same line encoding format as for MIMOSA26/Ultimate1, where encoded states/-clusters in a line are preceded by a 16-bit wide header. One also assumes an occupancy high enough to account for one line address header for every single line in the matrix, which for 1024 lines gives 1024 words.

The number of available words for storing hits then becomes **2x2048 - 1024 = 3072**. In addition comes the storage occupied due to fake hits.

One can now compare the storage capabilities with respect to the different types of coding:

[3]Cross shaped cluster with 3 pixels above threshold vertically and 3 hits above threshold horizontally

Encoding	**Formula**	**Result**
State encoding (two words for noisy pix.)	$\frac{(3072-37\cdot 2)\ words}{6\ words/hit}$	499 hits
Fixed-length clusterisation (three words for noisy pix.)	$\frac{3072-37\cdot 3\ words}{3\ words/hit}$	987 hits
Variable-length clusterisation (one word for each noisy pix.)	$\frac{(3072-37)\ words}{3\ words/hit}$	1011 hits
State encoding in high-res. substr. (max. two states pr hit)	$\frac{3072-37\cdot 2\ words}{4\ words/hit}$	749 hits
Variable-length clusterisation in high-res. substr. (max. two states pr hit)	$\frac{(3072-37)\ words}{2\ words/hit}$	1517 hits

Given a fixed memory size, there is a large difference between the different techniques in number of hits that can be stored, as illustrated in Figure E.5.

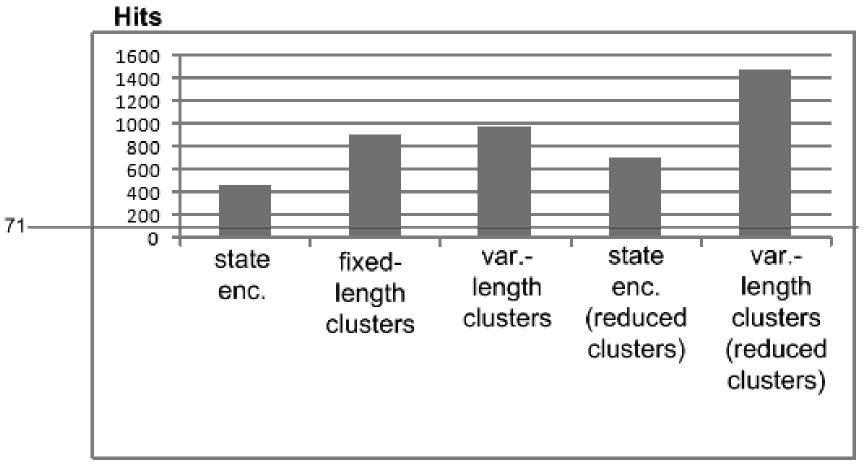


Figure E.5: Memory storage capabilities for the different encodings, given a fixed memory size of 2x2048 16-bit words.

The previous calculations are based on the assumption that there is a uniform distribution of hits throughout the entire matrix of the chip and no overflow situation. To find the absolute maximum of hits that are possible to store into memory, one could also start in the other end, with the extreme situation of each line containing 3 hits due to particles and no fake hits, which for the state encoding in a normal MAPS substrate would be the maximum number of hits per line before overflowing. The memory storage capabilities with 2x2048 words then becomes:

Encoding	**Formula**	**Result**
State encoding (3 states/cluster)	$\frac{2 \cdot 2048 \quad words}{19 \; words/line}$	215 lines (645 hits)
Fixed-/Variable length clusters	$\frac{2 \cdot 2048 \; words}{10 \; words/line}$	409 lines (1227 hits)
State encoding in high-res. (max 2 states/cluster)	$\frac{2 \cdot 2048 \; words}{13 \; words/line}$	315 lines (945 hits)
Var.-length clusters in high-res. (reduced. avg. cluster size)	$\frac{2 \cdot 2048 \; words}{(3 \cdot 2+1) words/line}$	585 lines (1755 hits)

The maximal number of hits to be stored in the memories can now be tabulated, as shown in Figure E.6.

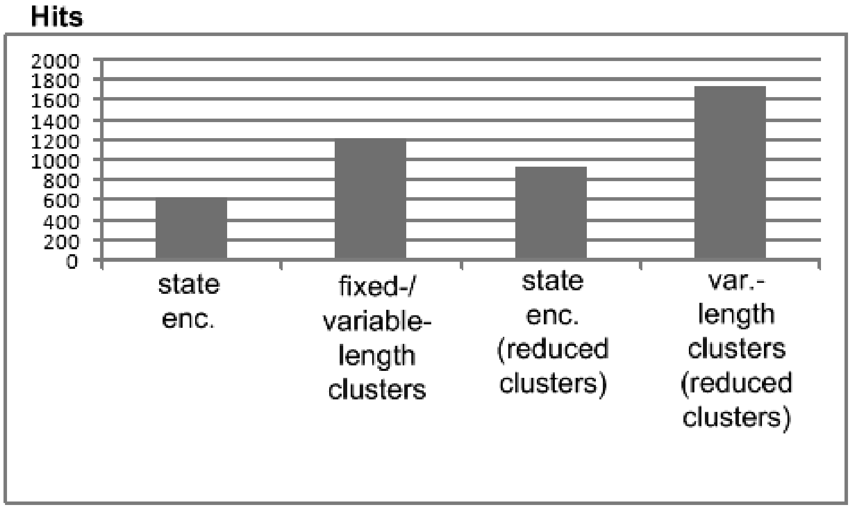


Figure E.6: Memory storage capabilities for the different encodings, assuming three hits due to particles in every line.

E.2.7 Conclusion

As long as the target for the experiment is ILC, which is supposed to have maximum 71 hits per chip, we note that all the considered cases meet the requirements in terms of memory storage.

However, the technique with variable-length clusterisation, and especially if smaller clusters are obtainable, optimizes the space in the memory.

Still, there are considerable challenges of implementing the cluster finder, as the currently implemented cluster finder consumes a large amount of area compared to the overall area of the pixel matrix. The clusterisation circuitry that is developed by X. Fang (targeted at a 5-bit ADC) is utilising a pipeline that monitors three lines at a time. Due to the large amount of flip flops utilised for this pipeline, the height of the cluster finder becomes as much as 1500 micrometers. While for comparison, the height of the entire digital readout electronics of MIMOSA26 is 2100 micrometers. As this circuitry is to be allocated in the periphery of the active detector chip area, a large amount of overlap between sensors is implied to obtain full fill factor, which comes at the expense of material budget.

Appendix F

Block schematics and implementation details for CBM 3D readout

This appendix contains the block schematics for some of the features of the proposed readout architecture for CBM [100].

F.1 Pixel cells

To perform the state encoding with two bits, the following two bits are encoded:

$$code(0) = disc(i+2) \cdot disc(i+1)$$
$$code(1) = [\overline{disc(i+2)} \cdot disc(i+2)] + [disc(i+3) \cdot disc(i+1)]$$

And the condition for a pixel to grab the token and pass the state information to the readout, is in its most simple form that

$$grab\_token = disc(i) \cdot \overline{disc(i-1)}$$

This logic works well as long as there are no line patterns of consecutive ones longer than four, and since the average length of a cluster is assumed to be two, this assumption should hold for the large majority of hits, but at the same time, one cannot exclude rare cases of overlapping clusters that case patterns longer than four pixels.

Two strategies for meeting these cases are proposed:

- The individual pixels continue to only see hits of nearest neigbours, but the number of neighbours seen to the right in increased from one to four. In addition to grabbing the token when the leftmost neighbor is zero, it also grabs the token when all the leftmost neighbors are one. This strategy implies redundant state encoding in the cases where the number of pixels in a pattern exceeds five, but as such cases are assumed rare, such redundancy can be considered acceptable.

- When an individual pixel indentifies itself as the individual pixel in a state, it will in the cases of a seeing fully occupied state, propagate a message to the pixel element four locations ahead that it must identify itself as a state if it contains hits. This way of identifying states results in nonredundant state encoding, but it also requires the propagation of a signal that theoretically can propagate through the entire line.

See Figure F.1.

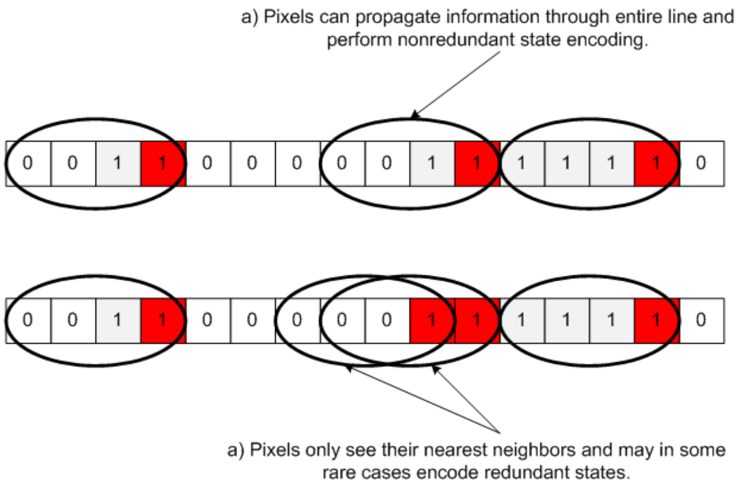

a) Pixels can propagate information through entire line and perform nonredundant state encoding.
0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 0
0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 0
a) Pixels only see their nearest neighbors and may in some rare cases encode redundant states.

Figure F.1: Redundant vs nonredundant state encoding. Pixels performing redundant encoding only see the discriminated hit signals of their nearest neighbors, while pixels performing nonredundant state encoding allow as long as necessary the propation of a supervision signal.

F.1.1 Redundant case, only seeing hits of nearest neighbors

The conditions for a hit pixel to grab the token are the two cases where either the rightmost neighbor is zero or where all the four rightmost neighbours are one:

$$grab\_token(i) = disc(i) \cdot [\overline{disc(i-1)} + disc(i-1) \cdot disc(i-2) \cdot disc(i-3) \cdot disc(i-4)]$$

Using De Morgan's theorems, this logic can be simplified. First, we write the statement to be AND'ed with the hit signal as

$$\overline{A} + A \cdot B, \ where\ A = disc(i-1), \ and\ B = disc(i-2) \cdot disc(i-3) \cdot disc(i-4).$$

And then, we first expand and then simplify the statement to:

$$(\overline{A} \cdot \overline{B} + \overline{A} \cdot B + \overline{A} \cdot B) + A \cdot B = \overline{A} \cdot (\overline{B} + B) + B \cdot (\overline{A} + A) = \overline{A} + B = \overline{A \ \cdot \ \overline{B}}$$

Reinserting for A and B, we arrive at:

$$\overline{disc(i-1) \ \cdot \ [\overline{[disc(i-2) \cdot disc(i-3) \cdot disc(i-4)}]}$$

So that the final equation for grabbing the token becomes

$$grab\_token(i) = disc(i)\ AND\ [\overline{disc(i-1) \ \cdot \ [\overline{[disc(i-2) \cdot disc(i-3) \cdot disc(i-4)]}]}$$

Since the result of the AND'ing of (i-2) and (i-3) already is given by the code(0) bit of pixel cell (i-4), we can write the new equation as:

$$grab\_token(i) = disc(i)\ AND\ [\overline{disc(i-1) \cdot [\overline{codebit\_zero(i-4) \cdot disc(i-4)}]}$$

F.1.2 Nonredundant case, propagating hit information as long as necessary

Instead of checking the discriminator value of the pixel four elements behind (disc(i-4)), we recursively check if this pixel element also has requested to grab the token:

$$grab\_token(i) = disc(i)\ AND\ [\overline{disc(i-1) \cdot [\overline{[codebit\_zero(i-4) \cdot grab\_token(i-4)]}]}].$$

Schematics of the implemented logic are shown in Figure F.2 and Figure F.3.

Compared to the original schematic, we see here that one inverter has been removed and two NAND gates have been added, so the increase in logic for implementing this function is minimal.

We also see that in a group of four pixels, there is a path for an enable enforcement signal going through two NAND gates and one AND gate. Although completely unlikely in practice, there is a theoretical possibility that there are hits

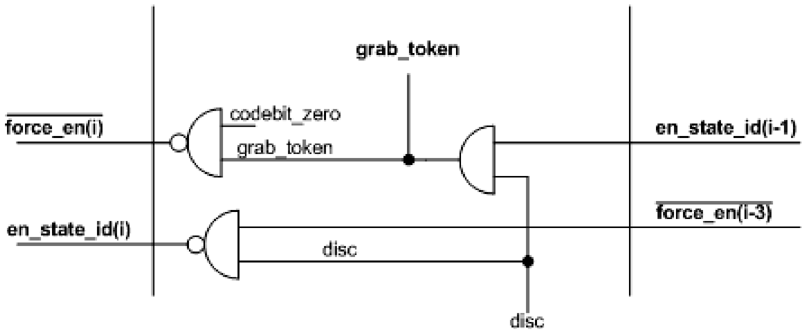


Figure F.2: Logic of pixel cell for state identification and propation of state identification conditions to next pixel four elements ahead.

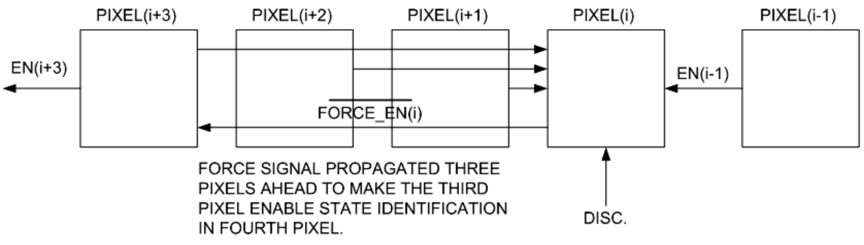

PIXEL(i+3)
PIXEL(i+2)
PIXEL(i+1)
PIXEL(i)
PIXEL(i-1)
EN(i+3)
EN(i-1)
FORCE_EN(i)
FORCE SIGNAL PROPAGATED THREE PIXELS AHEAD TO MAKE THE THIRD PIXEL ENABLE STATE IDENTIFICATION IN FOURTH PIXEL.
DISC.

Figure F.3: Organization of pixels with distributed and lossless, nonredundant state identification.

in each pixel in the line, and thus a force signal being propagated through the entire line:

The typical delay of an two-input AND of drive strength one is 0.080 ns, while the typical delay of a two-input NAND of drive strength one is 0.035 ns. Ingnoring wire delay, the propagated delay through one cell therefore becomes $(0.080\ ns + 2 \cdot 0.035 ns) = 0.15\ ns$. Since the signal passes through each fourth cell, the total number of cells to pass in the theoretical worst case becomes $1024/4 = 256$, and the total delay would in this case become 38.4 ns.

Even in the case of a fully occupied line, all signals should thus have stabilized 40 ns after the last line is processed by the rolling shutter. However, if the token and state idenfitication logic is propagated in the same direction, the first state can be read out even before all states in the line have been identified, as illustrated in Figure F.4.

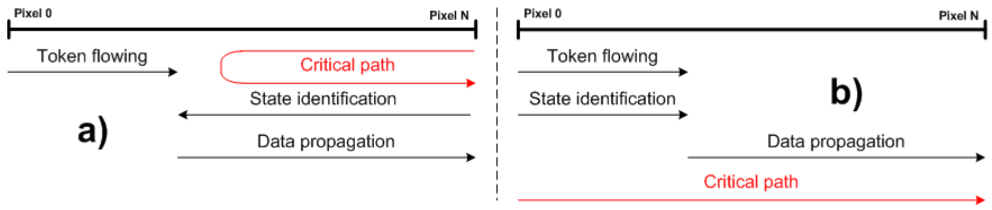


Figure F.4: The two possible signal paths for nonredundant state identification.

The solution with nonredundant state encoding was therefore chosen for implementation.

F.2 Line buffers

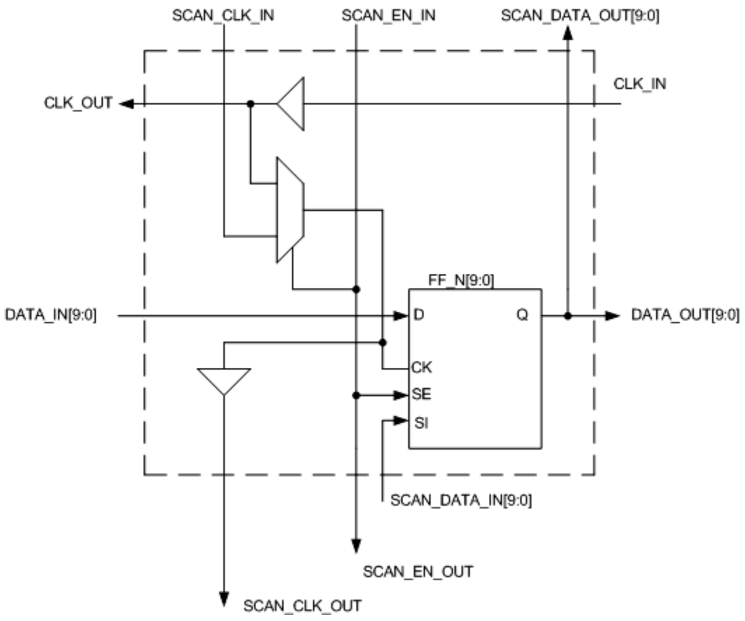


Figure F.5: Structure of 10-bit line buffer. Data going from the left to the right while clocked by the token clock, and going from top to bottom during scan mode.

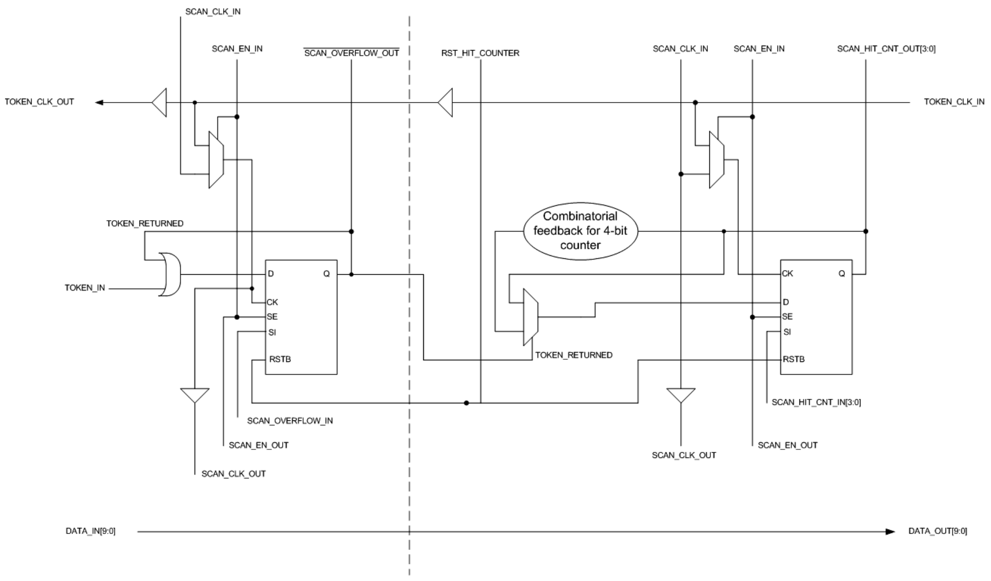


Figure F.6: Structure of line buffer hit counter. Hit counter is enabled until the token has returned.

Appendix G

Abbreviations

This appendix contains a list of abbreviations used in this thesis.

ADC	Analog / Digital Converter
ALICE	A Large Ion Collider Experiment
AOI	And / Or / Invert (type of standard cell)
ASIC	Application Specific Integrated Circuit
BEOL	Back End Of Line
CBM	Compressed Baryonic Matter (experiment)
CCE	Charge Collection Efficiency
CERN	European organization for nuclear research
CMOS	Complementary Metal Oxide Semiconductor
CNRS	Centre National de la Recherche Scientifique
CS	Chip Select
DAQ	Data Acquisition
DCT	Discrete Cosine Transform
DMA	Direct Memory Access
ECAL	Electromagnetical CALorimeter
ENC	Equivalent Noise Charge
EUDET JRA-1	EUropean DETector Joint Research Activity 1 (beam telescope)
FAIR	Facility for Antiproton and Ion Research (particle accelerator)
FEOL	Front End Of Line
FPGA	Field Programmable Gate Array
FIFO	First In First Out
IRFU	Institut de la Recherche sur les lois fundamentales de l'Univers
FSM	Finite State Machine
HEP	High Energy Physics
HLT	High Level Trigger
ILC	International Linear Collider
IPHC	Institut Pluridisciplinaire Hubert Curien
IO	Input / Output

JPEG	Joint Photographics Experts Group (consortium for image treatment)
JTAG	Join Test Action Group (consortium establishing current IEEE 1149.1 boundary scan standard)
LFSR	Linear Feedback Shift Register
LHC	Large Hadron Collider
LVDS	Low Voltage Differential Signaling
MAPS	Monolithic Active Pixel Sensors
MIMOSA	Minimum Ionizing Particle MOS Active Pixel Sensor (ASIC)
MIMOSIS-1	MIMOSA for SIS-100 (ASIC)
MIT	Massachusetts Institute of Technology
MIT LL	MIT Lincoln Labs
MUX	Multiplexer
MVD	Micro Vertex Detector
PCB	Printed Circuit Board
PCI	Peripheral Component Interconnect (bus standard)
PLA	Priority Look-Ahead
RAM	Random Access Memory
RICH	Ring Imaging CHerenkov (particle identification detector)
RLE	Run Length Encoding
SDS	Sparse Data Scan
SIS-100	Superconduction Ion Synchrotron 100 (heavy ion synchrotron used for CBM)
SLAC	Stanford Linear Accelerator Center
SLD	SLAC Linear Detector
SLID	Solid-Liquid-Interdiffusion
SNR	Signal-to-Noise Ratio
SOI	Silicon On Insulator
SPI	Serial Peripheral Interface bus
STAR	Solenoid Tracker At the Relativistic Heavy Ion Collider (experiment)

SUZE	SUppression of ZEroes (ASIC)
TID	Total Ionizing Dose
TOF	Time Of Flight
TRD	Transition Radiation Detector
TSV	Through Silicon Via
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit

List of Figures

1.1 Charge Sensitive Amplifier (CSA) with shaper. The detector sensing element is modeled as a reverse biased p-n diode with a parasitic capacitance C_D, while the shaper is modeled as a low pass filter followed by a high pass filter (RC-CR). 2

1.2 MAPS operation principle. A cross sectional view showing the charge diffusion to the left, and with the corresponding potential distribution of the two cross sections A and B shown to the right. 8

1.3 Distribution of collected charge for 5.9 keV photons [1]. 9

1.4 Block schematic of a basic MAPS pixel cell. 10

1.5 Simplified block diagram of a single array of a MIMOSA circuit [31]. 11

1.6 CBM detector setup. Left: Configuration for electrons. Right: Configuration for muon measurements [42]. 14

2.1 Particle passing through substrate of a planar pixel sensor. Side view to the left, showing the particle passing below different pads, and top view to the right, showing the clusterised manifestation in the pixel matrix. We also notice that one of the pixels in the cluster is neigbour to all the other pixels in the clusters - the central pixel. 23

2.2 a) Readout pixel by pixel (early MIMOSA chips). b) Readout line by line (MIMOSA26) 25

2.3 Block schematic of rolling shutter operated pixel matrix with data sparsification circuitry in the periphery. The preamplifiers of each line is powered on and the pixel signal read out at each their time slot. While the discriminators and the readout electronics in the periphery are shared for all the lines and always being active. 26

2.4 Combinatorial search for states within the state encoder. A sliding window of four pixels identifies groups of hits, called states. 26

2.5 State encoding of a regular cluster. 31

2.6 Block schematic of the state encoding circuitry [53]. 31

2.7 Timing waveform for the state encoding circuitry. 32

2.8 Decomposition of a cluster [53]. 34

2.9 Block schematic of the cluster finding and encoding circuitry [53]. 35
2.10 Compression efficiencies of state encoding (dash-dotted line) and clusterization (solid line) shown as functions of occupancy (ranging from zero to one percent). The occupancies of EUDET-JRA1, STAR 500 and ILC are inserted with separate markers. 36
2.11 Data rates with state encoding shown as functions of frame readout time for the fixed particle fluxes of EUDET-JRA1, STAR 500 and ILC. 38

3.1 Functional view of MIMOSA26. 50
3.2 Block diagram of pixel cell with interfaces 52
3.3 Waveform for analog pixel cell readout. 52
3.4 Discriminator and correlated double sampling operating principle. 53
3.5 Data path with sequencer for the readout and sparsification circuitry. Timing waveform for the control signals is shown in a) and the functional block schematic shown in b). 55
3.6 The shape of the CKLATCH signal is controlled with a 16-bit signal where each bit correspond to one small time interval in the signal period. 56
3.7 The 18 banks of sparse scan with state encoding. 56
3.8 Token blocking and token releasing by intertwined NAND/NOR cells in sparsified scan path. 57
3.9 Block schematics of NAND and NOR cells in the sparsified scan chain. 57
3.10 Multiplexer with two parallell paths merged in a second stage. 58
3.11 Functional block diagram of first stage multiplexer with token passing. 59
3.12 Data packet format for a processed line. 61
3.13 Schematic block diagram of memory management. 61
3.14 Odd/even swapping procedure. 62
3.15 Block diagram of serializer and output mux. 63
3.16 Initiation details of serial transfer. When the header of a new frame is sent, a synchronization strobe (marker) is asserted for the first four clock periods of the transmission. 64
3.17 Data format of serialization in mode 0 and 1. 64
3.18 Data format of serialization in mode 2 and 3. 64
3.19 Conceptual view of scan test system. 66
3.20 Data format for the three different scan tests. 67
3.21 General design verification setup. 69
3.22 Standalone datapath verification setup. 70
3.23 Verification setup for entire design with JTAG. 71
3.24 Top level of test bench for entire design with JTAG. STIM_CORE_DIGITAL gives all the stimuli and captures all the responses of the design. 71

3.25 Content of stimuli module (STIM_CORE_DIGITAL). 73
3.26 Simulation cycle for two-line pattern. 74
3.27 Reconstructed pixel matrix shown as a function of actual pixel matrix content. 79
3.28 Test setup of MIMOSA26 readout electronics. 81
3.29 a) Token passing handshake of SUZE multiplexer. b) Token passing handshake of MIMOSA26 multiplexer. 85
3.30 Timing waveform for addressing and writing data to odd and even memories. . 88
3.31 Block schematic of improved NAND/NOR cells in sparse scan chain. 89
3.32 Timing diagram of improved NAND/NOR cells in sparse scan chain. 90
3.33 MIMOSIS-1. Double-sided readout chip. Two matrices, each of dimensions 256x1152, are read out from each their side [63]. 92

4.1 Functional block schematic of a pixel cell in the 3-tier Fermilab demonstrator chip [72]. 97
4.2 Illustration of a Tezzaron two-tier stack which is about to be bounded with a third tier [79]. 102
4.3 Development flow for the ATLAS 3D pixels [83]. 103
4.4 Organisation of the first HEP Multiproject Wafer [85]. 104
4.5 Functional block schematic of a three tier design with shaperless frontend [71]. 105
4.6 Functional block schematic of the 3D integrated rolling shutter architecture. Each pixel has its separate discriminator, implemented at the second tier using a high-gain amplifier composed of four cascaded low-gain amplifiers. 106
4.7 Prototyping strategy for ILC 3D chip. 109
4.8 a) External interfaces of digital tier. b) Readout sequence of digital tier. . . . 109
4.9 Functional block schematic of column pair with gray counter. 110
4.10 Functional block schematic of the organisation of columns and their clock distribution. 111
4.11 Functional block schematic of the analog pixel cell. 111
4.12 Functional block schematic of the digital pixel cell. 112
4.13 Schematic block diagram of the readout control with embedded 8b10b encoder. External input pads are shown in red and external output pads are shown in blue. 115
4.14 Waveform showing how the RO_8b10b and TIER2CORE modules are synchronised with each others. 115
4.15 Behaviour of the state machine with embedded 8b10b encoder. 116
4.16 Synchronisation sequence of external test pads. 117
4.17 Test of the shift register chain. Data comes out after a latency of 24576 clock cycles, which are necessary to cross the shift register. 118
4.18 Emulation of particle hits using the READ, RDTEST and ROCLK signals. . . 118

4.19 Test setup for 3D chip. For this automatized test, one could for example use the Agilent 16822A that is both a pattern generator and a logic analyzer. 119
4.20 Waveform showing asynchronous readout. 120
4.21 Waveform showing synchronous readout. 120

5.1 a) Preliminary arrangement of sensors around the CBM beamhole. Alternative options may be proposed if found more useful [92]. b) Proposal of arrangement with 3D detectors having close to 100 % active surface. 124
5.2 Simulation results showing the density of hits per frame with 500,000 frames/second and 10^6 collisions/second. Green: $0.1 - 0.5$ $hits/mm^2/frame$. Yellow: $0.5 - 1.4$ $hits/mm^2/frame$. Orange: ≤ 1.5 $hits/mm^2/frame$. 125
5.3 Charge collection efficiency versus radiation dose for XFAB high-resistitivity MAPS [44]. 128
5.4 Rolling shutter operated pixel matrix organized in 10 submatrices. 131
5.5 Splitting of submatrix in two halves to make token injection compliant with rolling shutter operation. 133
5.6 Operation principle of distributed state encoding. Every pixel in a line (pixel i) communicates with its three preceding pixels (pixel (i+3), pixel (i+2), pixel (i+1)) to perform the state encoding. It also receives an enable signal from its first succeeding pixel (pixel (i-1)), allowing the pixel to grab the flowing token if not already done by the succeeding pixel. 134
5.7 State encoding circuitry of a pixel cell. 135
5.8 Circuitry for valid state identification and token-based data transmission in a pixel cell. 136
5.9 Since the address bits are fixed, the address multiplexer can be replaced by faster NAND and NOR gates. 136
5.10 Abstract view of pixel cell. 137
5.11 Organization of line buffers. Yellow: First digital tier. Blue: Second digital tier. 137
5.12 Critical path of a line segment of pixels. 139
5.13 Cross-section of 20 µm long metal 2 wire, surrounded by metal everywhere at smallest possible spacing to provide worst-case reference for wire delay. 141
5.14 The line buffers can be operated in acquisition mode and in scan mode. . . . 142
5.15 Readout of upper and lower half of a submatrix, controlled by a common FSM. The FSM performs pipelined readout of the buffers in one half while the token passing circuitry provides new content to the buffers in the other half. 143
5.16 State selection mechanism of the readout FSM. For each clock cycle, the state counters of the selected segments are decremented and the content of the buffers is shifted one or two places. The buffers that are seen by the readout FSM are shown in gray. 144

5.17 Floorplan of the second digital tier with the column buffers allocated (top view). 147
5.18 Possible floorplan of second digital tier with the column buffers and readout structures allocated. 147
5.19 3D stack of the entire architecture. 147
5.20 A folded SUZE with the zero suppression circuitry on top of the sensor, avoiding the dead area in the periphery. 157

D.1 Two-line test pattern for critical path activation. 178
D.2 Two-line test pattern for testing left half of matrix. 179
D.3 Two-line test pattern for testing right half of matrix. 180

E.1 a) Two separate branches in the analog circuitry. b) Binary discrimination performed in digital circuit following ADC. 186
E.2 Data format of encoded state with ADC data. 186
E.3 Shape of an encoded cluster. 187
E.4 Data formatting of encoded clusters. a) Encoded cluster of variable length. b) Encoded cluster of fixed length. 187
E.5 Memory storage capabilities for the different encodings, given a fixed memory size of 2x2048 16-bit words. 190
E.6 Memory storage capabilities for the different encodings, assuming three hits due to particles in every line. 191

F.1 Redundant vs nonredundant state encoding. Pixels performing redundant encoding only see the discriminated hit signals of their nearest neighbors, while pixels performing nonredundant state encoding allow as long as necessary the propation of a supervision signal. 194
F.2 Logic of pixel cell for state identification and propation of state identification conditions to next pixel four elements ahead. 196
F.3 Organization of pixels with distributed and lossless, nonredundant state identification. 196
F.4 The two possible signal paths for nonredundant state identification. 197
F.5 Structure of 10-bit line buffer. Data going from the left to the right while clocked by the token clock, and going from top to bottom during scan mode. 198
F.6 Structure of line buffer hit counter. Hit counter is enabled until the token has returned. 199

Bibliography

[1] R. Turchetta, J.D. Berst, B. Casadei, G. Claus, C. Colledani, W. Dulinski, Y. Hu, D. Husson, J.P. Le Normand, J.L. Riester, G. Deptuch, U. Goerlach, S. Higueret, and M. Winter. A monolithic active pixel sensor for charged particle tracking and imaging using standard VLSI CMOS technology. *Nuclear Instruments and Methods in Physics*, 458:677–689, 2001.

[2] C. Hu-Guo, J. Baudot, G.Bertolone, A. Besson, A.S. Brogna, C.Colledani, G. Claus, R. DeMasi, Y. Degerli, A. Dorokhov, G. Doziere, W. Dulinski, X. Fang, M. Gelin, M. Goffe, F. Guilloux, A. Himmi, K. Jaaskelainen, M. Koziel, F. Morel, F. Orsini, M. Specht, Q. Sun, O. Torheim, I. Valin, and M. Winter. *First reticule size MAPS with digital output and integrated zero suppression for the EUDET-JRA1 beam telescope.* Nuclear Instruments and Methods in Physics Research, 2010. doi:10.1016/j.nima.2010.03.043.

[3] L. Rossi, P. Fischer, T. Rohe, and N. Wermes. *Pixel detectors. From fundamentals to applications.* Springer Verlag, first edition, 2006.

[4] J.M. Heuser, M. Deveaux, C. Muntz, and J. Stroth. Requirements for the silicon tracking system of CBM at FAIR. *Nuclear Instruments and Methods in Physics Research*, 568:258–262, 2006.

[5] A. Besson, C. Gilles, C. Colledani, Y. Degerli, G. Deptuch, M. Deveaux, W. Dulinski, N. Fourches, M. Goffe, D. Grandjean, F. Guilloux, S. Heini, A. Himmi, C. Hu, K. Jaaskelainen, Y. Li, P. Lutz, F. Orsini, M. Pellicioli, E. Scopelliti, A. Shabetai, M. Szelezniak, I. Valin, and M. Winter. A vertex detector for the International Linear Collider based on CMOS sensors. *Nuclear Instruments and Methods in Physics Research*, 568:233–239, 2006.

[6] G. Deptuch, A. Besson, G. Claus, C. Colledani, M. Deveaux, W. Dulinski, A. Gay, G. Gaycken, Yu. Gornushkin, D. Grandjean, A. Himmi, Ch. Hu, I. Valin, and M. Winter. Monolithic active pixel sensors adapted to future

vertex detector requirements. *Nuclear Instruments and Methods in Physics Research*, 535:366–369, 2004.

[7] Z.Y. Chang and W.M.C. Sansen. *Low-Noise Wide-Band Amplifiers in Bipolar and CMOS Technologies*. Kluwer Academic Publishers, first edition, 1991.

[8] A.S. Brogna, S. Buzzetti, W. Dabrowski, T. Fiutowski, B. Gebauer, M. Klein, C.J. Schmidt, H.K. Soltveit, R. Szczygiel, and U. Trunk. N-XYTER, a CMOS read-out ASIC for high resolution time and amplitude measurements on high rate multi-channel counting mode neutron detectors. *Nuclear Instruments and Methods in Physics Research*, 568:301–308, 2006.

[9] M. Turala. Silicon tracking detectors - historical overview. *Nuclear Instruments and Methods in Physics Research*, 541:1–14, 2005.

[10] E. Nygaard. *ASICs and electronic systems for readout of radiation-sensitive semiconductor detectors*. PhD thesis, University of Oslo, 1997.

[11] T. Kondo, R. Apsimon, G.A. Beck, P. Bell, R. Brennere, P. Bruckman de Renstromf, A.A. Carter, J.R. Carter, D. Charlton, W. Dabrowski, O. Dorholt, T. Ekelof, L. Eklund, M. Gibson, S. Gadomski, A. Grilloj, J. Grosse-Knetter, C. Haber, K. Haram, J.C. Hill, Y. Ikegami, Y. Iwata, L.G. Johansen, T. Kohriki, A. Macpherson, S. McMahon, G. Moorhead, J. Morin, J. Morris, M. Morrissey, K. Nagai, I. Nakano, J. Paterr, H. Pernegger, E. Perrins, P. Phillips, D. Robinson, B. Skubic, N. Spencer, S. Stapnes, B. Stugu, R. Takashima, S. Terada, M. Tyndel, N. Ujiie, Y. Unno, and M. Vos. Construction and performance of the ATLAS silicon microstrip barrel modules. *Nuclear Instruments and Methods in Physics Research*, 485:27–42, 2002.

[12] A. Ahmad et al. The silicon microstrip sensors of the ATLAS semiconductor tracker. *Nuclear Instruments and Methods in Physics Research*, 578:98–118, 2007.

[13] J. Grosse-Knetter. The ATLAS pixel detector. *Nuclear Instruments and Methods in Physics*, 549:70–74, 2005.

[14] Y. Unno. Silicon sensor upgrade for the ATLAS upgrade for SLHC. *Nuclear Instruments and Methods in Physics Research*, 569:41–47, 2006.

[15] J. Kemmer and G. Lutz. New detector concepts. *Nuclear Instruments and Methods in Physics Research*, 253:365–377, 1987.

[16] H. Kruger. *Front-end electronics for DEPFET pixel detectors at SuperBelle (BELLEII).* Nuclear Instruments and Methods in Physics Research, 2009. doi:10.1016/j.nima.2009.10.042.

[17] J.J. Velthuis et al. DEPFET, a monolithic active pixel sensor for the ILC. *Nuclear Instruments and Methods in Physics Research*, 579:685–689, 2007.

[18] P. Fischer et al. Progress towards a large area, thin DEPFET detector module. *Nuclear Instruments and Methods in Physics Research*, 582:843–848, 2007.

[19] S.I. Parker, C.J. Kenney, and J. Segal. 3D - a proposed new architecture for solid-state radiation detectors. *Nuclear Instruments and Methods in Physics Research*, 395:328–343, 1997.

[20] A. Pozza, M. Boscardin, L. Bosisio, G.-F.D. Betta, C. Piemonte, S. Ronchin, and N. Zorzi. First electrical characterization of 3D detectors with electrodes of the same doping type. *Nuclear Instruments and Methods in Physics Research*, 570:317–321, 2006.

[21] S.I. Parker, C.J. Kenney, D. Gnani, A.C. Thompson, E. Mandelli, G. Meddeler, J. Hasi, J. Morse, , and E.M. Westbrook. 3DX: An x-ray pixel array detector with active edges. *IEEE Transactions on Nuclear Science*, 53(3):1676–1688, 2006.

[22] P. Hansson. *3D Silicon Pixel Sensors. Recent Test Beam Results.* Vienna Conference on Instrumentation, Vienna, February, 2010.

[23] M. Lozano, E. Cabruja, A. Collado, J. Santander, and M. Ullan. Bump bonding of pixel systems. *Nuclear Instruments and Methods in Physics*, 473:95–101, 2001.

[24] Ch. Broennimann, F. Glaus, J. Gobrecht, S. Heising, M. Horisberger, R. Horisberger, H. C. Kastli, J. Lehmann, T. Rohe, and S. Streuli. Development of an indium bump bond process for silicon pixel detectors at PSI. *Nuclear Instruments and Methods in Physics*, 565:303–308, 2006.

[25] C.J. Kenney, S.I. Parker, I. Sherwood, V.Z. Peterson, W.J. Snoeys, J.D. Plummer, and H.A. Chye. A prototype monolithic pixel detector. *Nuclear Instruments and Methods in Physics Research*, 342:59–77, 1994.

[26] C.J.S. Damerell. CCD-based vertex detectors. *Nuclear Instruments and Methods in Physics*, 541:178–188, 2005.

[27] P. C. Rowson and D. Su. Highlights of the SLD physics program at the SLAC linear collider. *Annual Review of Nuclear and Particle Science*, 51:345–412, 2001.

[28] S. Worm. *Recent results with CCDs for the Linear Collider.* Presentation at VERTEX-2006. Perugia, September 25th to 29th, 2006.

[29] G. Deptuch. *New Generation of Monolithic Active Pixel Sensors for Charged Particle Detection.* PhD thesis, Universite Louis Pasteur Strasbourg, 2002.

[30] G. Deptuch, G. Claus, C. Colledani, M. Deveaux, A. Gay, W. Dulinski, Yu. Gornushkin, Ch. Hu-Guob, and M. Winter. Development of monolithic active pixel sensors for charged particle tracking. *Nuclear Instruments and Methods in Physics*, 511:240–249, 2003.

[31] R. Turchetta et al. *A monolithic active pixel sensor for charged particle tracking and imaging using standard VLSI CMOS technology.* Nuclear Science Symposium Conference Record, 2004.

[32] W. Dulinski, J. Baudot, A. Besson, G. Claus, C. Dritsa, M. Goffe, M. Winter, and A. Zarnecki. *Beam Telescope for Medium Energy Particles based on Thin, Submicron Precision MAPS.* Proceedings paper at Nuclear Science Symposium 2007, 2007.

[33] G. Traversi, A. Bulgheroni, M. Caccia, M. Jastrzab, M. Manghisoni, E. Pozzati, L. Ratti, and V. Re. First generation of deep n-well CMOS MAPS with in-pixel sparsification for the ILC vertex detector. *Nuclear Instruments and Methods in Physics*, 604:390–392, 2009.

[34] V. Re. Status and perspectives of deep n-well 130 nm CMOS MAPS. *Journal of Instrumentation*, 3:3–5, 2009.

[35] N. Neri et al. *Deep n-well MAPS in a 130 nm CMOS technology: Beam test results.* Nuclear Instruments and Methods in Physics Research, 2010. doi:10.1016/j.nima.2010.02.193.

[36] C. Hu-Guo. *Design and Characterisation of a Fast Architecture Providing Zero Suppressed Digital Output Integrated in a High Resolution CMOS Pixel Sensor for the STAR Vertex Detector and the EUDET Beam Telescope.* Topical Workshop on Electronics for Particle Physics, Naxos, September 15th to 19th, 2008.

[37] A. Dorokhov, G. Bertolone, J.Baudot, A.S. Brogna, C. Colledani, G. Claus, R. DeMasi, M. Deveaux, G. Doziere, W.Dulinski, J.-C. Fontaine, M. Goffe, A.Himmi, Ch. Hu-Guo, K. Jaaskelainen, M. Koziel, F. Morel, C.Santos, M. Specht, I. Valin, G. Voutsinas, F. M. Wagner, and M. Winter. *Improved radiation tolerance of MAPS using a depleted epitaxial layer.* Proceedings of the 11th European Symposium on Semiconductor Detectors,, 2009.

[38] A. Besson and M. Winter. *Status of the development of MIMOSA CMOS sensors.* Proceedings of the 2007 Internation Linear Collider Workshop,, 2007.

[39] M. Deveaux. *Development of fast and radiation hard Monolithic Active Pixel Sensors (MAPS) optimized for open charm meson detection with the CBM - vertex detector.* PhD thesis, University of Frankfurt am Main and Universite Louis Pasteur Strasbourg, 2008.

[40] R. Yarema. *3D and SOI Technology for Future Pixel Detectors.* Presentation at Common ATLAS CMS Electronics Workshop. CERN, March 19th to 21st, 2007.

[41] F. Khalid. *Frontend Electronics in SOI.* Presentation at VIPS-2010, Venice, April, 2010.

[42] V. Friese. *The CBM Experiment at FAIR.* Proceedings of the 4th International Workshop on Critical Point and Onset of Deconfinement,, 2007.

[43] I. Peric, L. Blanquart, G. Comes, P. Denes, K. Einsweiler, P. Fischer, E. Mandelli, and G. Meddeler. The FEI3 readout chip for the ATLAS pixel detector. *Nuclear Instruments and Methods in Physics Research*, 565:178–187, 2006.

[44] C. Hu-Guo. *Achievements and Perspectives of MIMOSA Sensors (MAPS) for Vertexing Applications.* Presentation at Vertex-2009. Veluwe, Sept. 13th to 18th 2009, 2009.

[45] http://www.eudet.org/.

[46] http://www.star.bnl.gov/.

[47] R. De Masi. *CMOS Pixel Sensors for High Precision Beam Telescopes and Vertex Detectors.* Presentation at 11th ICATPP Conference, October 5th to 9th, 2009.

[48] T. Alt, G. Grastveit, H. Helstrup, V. Lindenstruth, C. Loizides, D. Roehrich, B. Skaali, T. Steinbeck, R. Stock, H. Tilsner, K. Ullaland, A. Vestboe, T. Vik,

and A. Wiebalck. The ALICE high level trigger. *Journal of Physics G: Nuclear and Particle Physics*, 30, 2004.

[49] H.G. Essel. FutureDAQ for CBM: on-line event selection. *IEEE Transactions on Nuclear Science*, 53:677–681, 2006.

[50] I. Peric, L. Blanquart, G. Comes, P. Denes, K. Einsweiler, P. Fischer, E. Mandelli, and G. Meddeler. Dead-time free pixel readout architecture for ATLAS front-end IC. *IEEE Transactions on Nuclear Science*, 46:166–170, 1999.

[51] X. Llopart, M. Campbell, D. San Segundo, E. Pernigotti, and R. Dinapoli. *Medipix2, a 64k pixel read out chip with 55 um square elements working in single photon counting mode.* 2001 IEEE Nuclear Science Symposium Conference Record, 2002.

[52] V. Re. Status and perspectives of deep n-well 130 nm CMOS MAPS. *Journal of Instrumentation*, 3:3–5, 2009.

[53] X. Fang. Etude d'algorithmes de traitement d'images en vue de son implementation sur un ASIC. Master's thesis, 2007.

[54] J. Baudot. *Resultats de l'analyse des donnes du faisceau test SPS de MIMOSA 26.* IPHC internal note, 2009.

[55] M. Goffe. *Frequency study of MIMOSA26.* IPHC internal note, 2010.

[56] C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[57] A. Willig. *A short introduction to queueing theory.* Technical University Berlin, 1999.

[58] M. Winter. *Development of Swift and Slim CMOS Sensors for a Vertex Detector at the International Linear Collider.* ILC VD Review / ALCPG-07, Chicago, FNAL, November, 2007.

[59] G. Claus, C. Colledani, W. Dulinski, D. Husson, R. Turchetta, J.L. Riester, G. Deptuch, G. Orazi, and M. Winter. Particle tracking using CMOS monolithic active pixel sensor. *Nuclear Instruments and Methods in Physics Research*, 465:120–124, 2001.

[60] Y. Degerli, N. Fourches, M. Rouger, and P. Lutz. Low-power autozeroed high-speed comparator for the readout chain of a CMOS monolithic active

pixel sensor based vertex detector. *IEEE Transactions on Nuclear Science*, 50:1709–1717, 2003.

[61] Y. Degerli, N. Fourches, M. Rouger, and P. Lutz. Low-power autozeroed high-speed comparator for the readout chain of a CMOS monolithic active pixel sensor based vertex detector. *IEEE Transactions on Nuclear Science*, 53:3949–3955, 2006.

[62] H. Cui, S.C. Seth, and S.K. Mehta. Modeling fault coverage of random test patterns. *Journal of Electronic Testing: Theory and Applications*, 19:271–284, 2003.

[63] M. Winter. *CMOS Pixel Sensors for Charged Particle Tracking. Achieved Performances and Perspectives.* Presentation at TIPP-09, Tsukuba, March, 2009.

[64] P. Garrou, C. Bower, and P. Ramm. *Handbook of 3D Integration.* Wiley, 2007.

[65] G. Poupon. *Technical approach for 3D integration.* Presentation at 3D Integrated Technologies Perspectives. First workshop on LHC-ILC prospects. Paris, Nov. 29th, 2007.

[66] P. Siblerud. *Status and plans in 3D Technologies; summary - EMC-3D.* Presentation at 3D Integrated Technologies Perspectives. First workshop on LHC-ILC prospects. Paris, Nov. 29th, 2007.

[67] H. Sun, J. Liu, R.S. Anigundi, N. Zheng, J.-Q. Lu, K. Rose, and T. Zhang. 3D DRAM design and application to 3D multicore systems. *IEEE Design and Test of Computers*, 26:36–47, 2009.

[68] C.C. Liuand I. Ganusov, M. Burtscher, and S. Tiwari. Bridging the processor-memory performance gap with 3D IC technology. *IEEE Design and Test of Computers*, 22:556–564, 2005.

[69] B. Patti. *Vertical integration with Chartered Semiconductor and Tezzaron.* 7th International Meeting on Front End Electronics. Montauk, May, 2009.

[70] R. Yarema. *Development of 3D Integrated Circuits for HEP.* FERMILAB-PUB-06-343-E, 2006.

[71] W. Dulinski. *Thin, Fully Depleted Monolithic Active Pixel Sensor with Binary Readout based on 3D Integration of Heterogeneous CMOS Layers.* Presentation at TWEPP-2009. Paris, Sept. 23rd, 2009.

[72] R. Yarema. *Technical approach for 3D integration.* Common ATLAS CMS Electronics Workshop. CERN, March 19th-21st, 2007.

[73] R. Yarema. *3D Circuit Integration for Vertex and Other Detectors.* The 16th International Workshop on Vertex detectors, 2007.

[74] G. Deptuch. *Front End Electronics using 3D Integrated Circuits.* Presentation at FEE2009. Montauk, May 18th to 21st 2009, 2009.

[75] R. Yarema. *Review of 3D Related Technologies for HEP.* Presentation at 3D Integrated Technologies Perspectives. First workshop on LHC-ILC prospects. Paris, Nov. 29th, 2007.

[76] M. Demarteau. *3D Integrated Technology at Fermilab. Activities and Plans.* Presentation at 3D Integrated Technologies Perspectives. First workshop on LHC-ILC prospects. Paris, Nov. 29th, 2007.

[77] P. Enquist. *Direct Bond Interconnect.* 7th International Meeting on Front End Electronics. Montauk, May, 2009.

[78] A. Klumpp, R. Merkel, P. Ramm, J. Weber, and R. Wieland. Vertical system integration by using inter-chip vias and solid-liquid interdiffusion bonding. *Japanese Journal of Applied Physics*, 43, 2004.

[79] S. Gupta, M. Hilbert, S. Hong, and R. Patti. *Techniques for Producing 3D ICs with High-Density Interconnect.* Proceedings of the 21st International VLSI Multilevel Interconnect Conference, 2004.

[80] M. Motoyoshi. *SOI pixel detector with micro-bumps.* Presentation at VIPS-2010, Venice, April, 2010.

[81] M. Barbero, D. Arutinov, R. Beccherle, G. Darbo, R. Elyc, D. Fougeron, M. Garcia-Sciveres, D. Gnani, T. Hemperek, M. Karagounis, R. Kluit, V. Kostioukhine, A. Mekkaoui, M. Menouni, and J.-D. Schipper. A new ATLAS pixel front-end IC for upgraded LHC luminosity. *Nuclear Instruments and Methods in Physics Research*, 604:397–399, 2009.

[82] A. Macchiolo. *Application of a New Interconnection Technology for the ATLAS Pixel Upgrade at SLHC.* Presentation at TWEPP-2009. Paris, Sept. 23rd, 2009.

[83] S. Godiot et al. *3D electronics for hybrid pixel detectors.* Proceedings paper at TWEPP-09, 2009.

[84] R. Yarema. *The first vertically integrated MPW run for HEP.* Presentation at TWEPP-2009. Paris, Sept. 23rd, 2009.

[85] R. Yarema. *Presentation at the first Tezzaron 3D collaboration meeting.* FNAL Microelectronics Group, 2008.

[86] W. Dulinski, G. Bertolone, R. de Masi, Y. Degerli, A. Dorokhov, F. Morel, F. Orsini, L. Ratti, C. Santos, V. Re, X. Wei, and M. Winter. *Thin, Fully Depleted Monolithic Active Pixel Sensor based on 3D Integration of Heterogeneous CMOS Layers.* Proceedings paper at Nuclear Science Symposium 2009, 2009.

[87] W. Dulinski. *Ultra Thin, Fully Depleted MAPS based on 3D Integration of Heterogeneous CMOS Layers (3 Tiers).* 7th International Meeting on Front End Electronics. Montauk, May, 2009.

[88] Y. Degerli. *A vertically integrated rolling shutter binary MAPS with in-pixel digital memory.* Presentation at 3D meeting in Marseille, March 18th to 19th, 2010.

[89] M. Winter. *Occupancy of ILC-VD Inner Layer integrated over a complete bunch train.* IPHC internal note, 2008.

[90] A. Widmer and P. Franaszek. A DC-Balanced, partitioned-block, 8b/10b transmission code. *IBM Journal of Research and Development*, 27:440–451, 1983.

[91] M. Deveaux, S. Amar-Youcef, C. Dritsa, I. Frohlich, C. Muntz, S. Seddiki, J. Stroth, T. Tischler, and C. Trageser. *Design considerations for the Micro Vertex Detector of the Compressed Baryonic Matter experiment.* Proceedings paper at Vertex-2008, 2008.

[92] S. Seddiki. *MVD DAQ Prototype.* Presentation at 14th CBM Collaboration Meeting, Split, October, 2009.

[93] M. Deveaux. *Status of the Micro Vertex Detector of the Compressed Baryonic Matter Experiment.* Article submitted to Proceedings of Science, 2010.

[94] Y. Degerli. *Fast Binary Readout Monolithic Active Pixel Sensors: From CMOS to 3D-IT.* Presentation at FEE2009. Montauk, May 18th to 21st, 2009.

[95] D. Solomon. *Data compression. The complete reference.* Springer, 2007.

[96] R. Hashemian. Memory efficient and high-speed search Huffman coding. *IEEE Transactions on Communications*, 43:2576–2581, 1995.

[97] S. B. Choi and M. H. Lee. High speed pattern matching for a fast Huffman decoder. *IEEE Transactions on Consumer Electronics*, 41:97–103, 1995.

[98] M. Dahoumane. *Conception, Realisation et Caracterisation de l'Electronique, Integre de Lecture et de Codage des Signaux des Detecteurs de Particules Charges a Pixels Actifs en Technologie CMOS*. PhD thesis, University of Strasbourg, 2009.

[99] M. Winter. *Data flow delivered by a Vertex Detector made of CMOS sensors adapted to the ILC physics goals and running conditions*. IPHC internal note, 2006.

[100] GSI mbH. *An International Accelerator Facility for Beams of Ions and Antiprotons. Conceptual Design Report*. Gesellschaft fur Schwerionenforschung mbH, 2001.