Focal Plane Video Compression
Final Report

Submitted to Catalyst Foundation

Walter D. León-Salas
Department of Electrical Engineering
University of Nebraska-Lincoln

Advisors: Dr. Sina Balkir and Dr. K. Sayood

December 11, 2006

1 Introduction

This report summarizes the research work conducted at the University of Nebraska-Lincoln and funded by the Catalyst foundation under the grant "Focal plane video compression". The main motivation for this research work was to increase performance and reduce costs on digital cameras. Commercial implementations of image compression solutions require dedicated custom chips in addition to the sensor chip. By integrating a data compression scheme on the sensor chip increased performance and reduced costs are expected. Compressing the acquired images before being read out reduces the read out bandwidth requirements, thus, increasing performance. Implementing a data compression scheme on the sensor chip avoids the need of external custom chips, thus, reducing costs.

Standard CMOS processes offer the advantage of being able to integrate analog, digital, or mixed signal processing and computation circuits on the same silicon chip with the sensor [1], [2]. However, the photo-detectors available in standard CMOS processes suffer from lower signal-to-noise ratio (SNR), lower dynamic range (DR) and higher fixed pattern noise (FPN). These limitations kept CMOS photo-detectors from achieving performance similar to the CCD's. CMOS sensors were mainly used for low-precision, low-cost machine vision where image quality was not the driving factor. It was in the early 1990's that CMOS imaging technology was propelled by the advent of the Active Pixel Sensor (APS) technology. The APS provided the required SNR and dynamic range improvement by integration of an active amplifier into each pixel. FPN was also greatly reduced by employing Correlated Double Sampling (CDS) techniques. CMOS imaging technology has evolved ever since to the point that it now rivals the performance of CCD imagers.

In [3] a conditional replenishment technique is implemented. This is a lossy compression technique and works only for video sequences and not for still imaging. It reads out the pixels that differ from their value in the previous frame by more than a threshold. If the threshold is large, greater compression ratios can be achieved but at the cost of video quality. On the other hand, if the threshold is small the quality improves but so does the rate. This coding scheme is suited for high frame rate applications. The advantage of this approach is that it requires simple hardware to implement and can be done in a parallel fashion at the pixel level. A 32×32 imager has been demonstrated in a 2 μm CMOS process. Another approach reported in [4] employs the Discrete Cosine Transform (DCT). The computation of the DCT coefficients is performed in the analog domain with a switched capacitor circuit. The chip includes an array of 128×128 pixels, a two-dimensional DCT processor, and an analog-to-digital converter (A/D) that performs variable quantization. The DCT coefficients are computed for every non-overlapping block of 8 $\times$ 8 pixels. Subsequently, these coefficients are quantized and entropy coded. The entropy encoder is not included in the chip due to its complexity.

A 256 $\times$ 256 CCD image sensor that can be used as a front end for predictive coding is presented in [5]. Predictive coding is a technique employed in lossless image compression. It works by predicting the value of a pixel based on the value of its neighbors and encoding their difference. The approach in [5] reorganizes the pixels during read out so that the difference between a pixel and its predicted value is available at the output of the chip. The entropy coding stage is not included in the chip.

One of the challenges that a focal plane video compression system faces is the integration of the entropy coder on the same chip with the sensor. The large complexity of common entropy coding schemes like Huffman and arithmetic coding precludes this integration. Thus, designers have opted to implement only some of the image compression building blocks such as the DCT, predictive or wavelet decomposition on the focal plane and perform the entropy coding outside the focal plane, possibly by a dedicated custom chip or a general-application microprocessor. Because the DCT, pixel prediction or wavelet decomposition stages do not provide compression by themselves, placing the entropy coder outside the chip defeats the original purpose of decreasing the read-out bandwidth of the imager.

In this research work we addressed the problem of focal plane video compression and proposed a complete solution that integrates the image sensor, a pixel prediction circuit, an analog-to-digital converter (A/D) and an adaptive entropy coder on the same chip. The output of the integrated circuit is a compressed bit stream. This level of integration is possible due to the following contributions.

- Design, fabrication and test of an analog prediction circuit. The prediction circuit is presented and analyzed in Section 2 and the test results are presented in Section 5.

- Design, fabrication and test of a single-slope A/D with Golomb-Rice output codes. A single-slope integrating A/D is combined with a low-complexity Golomb-Rice entropy encoder by noticing that they have common circuitry. This combination enables a joint quantization/coding operation which reduces the circuit complexity. The design is presented in Section 2 and the measurements results are presented in Section 5.

- Design, fabrication and test of a full-scale 80$\times$44-elemet imager verifying the fundamental ideas. The imager architecture is presented in Section 2. The final chip was fabricated in a 0.35-μm CMOS technology and occupies a silicon area of 2.596 mm $\times$ 5.958 mm. Acquired images along with the imager compression performance are presented in Section 5.

- The design of two additional (dual-slope and cyclic) A/D with Golomb-Rice output. The dual-slope A/D was tested at the transistor level. The cyclic A/D was tested at the system level. The designs are presented in Section 4.
- Conception, analysis, implementation and verfication of an adaptive low-complexity selection of the Golomb-Rice coding parameter. Despite its low complexity this algorithm performs remarkably well and has a compression performance comparable to more sophisticated entropy coders. The adaptive selection is reported in Section 5.

The contributions of this research work are not limited to only focal plane video compression. The possibility of performing an adaptive joint quantization and coding opens new and exciting opportunities in sensor networks applications where low cost and low complexity compression solutions will enhance the functionality of sensor nodes.

2 Imager Design

The chip design follows a column-parallel architecture. The advantage of column-parallel architectures is that they relax the speed constraints of the analog-to-digital converters. Since each column has a dedicated A/D, each A/D has to operate only at the row rate [6]. It has also the advantage of enabling a sequential conversion and readout. The disadvantage is that the A/D form-factor has to match that of the pixel pitch [7]. In cases where the pixel pitch is too small, an A/D can be shared by a group of four, eight or ten columns [7], [8].

Figure 1 depicts a block diagram of the architecture employed in this work. By choosing a column-level architecture, the A/Ds do not have to run at high speeds. Therefore, slower converters like single-slope integrating A/Ds can be employed. The advantage of the integrating converters is that they can be mixed with a low complexity entropy encoder. The chip consists mainly of an 80$\times$44 array of pixels, a bank of 44 column processors, and control logic.

A block diagram of the column processor is shown in Figure 2. Each processor is composed of an analog memory bank, a correlated-double-sampling (CDS) circuit, an analog pixel prediction circuit, a single-slope A/D, an adaptive Golomb-Rice entropy encoder, and read-out logic. There is one processor for each pixel column. The analog

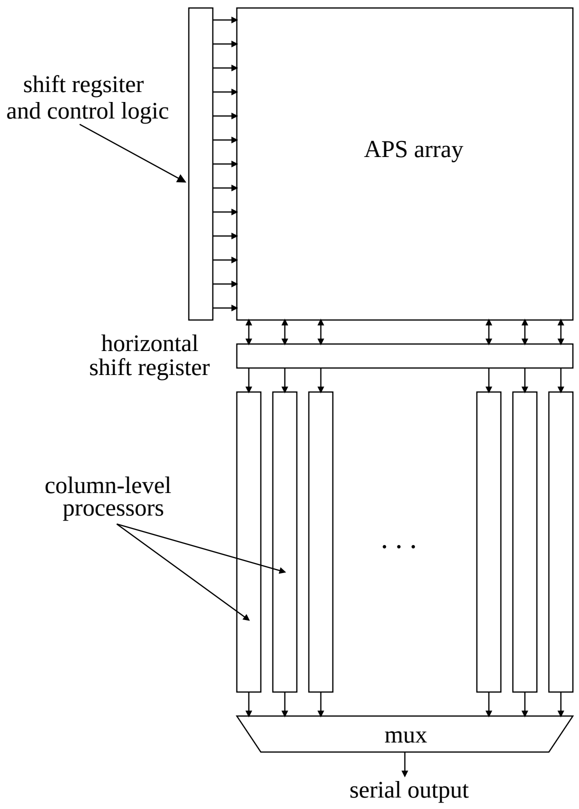


Figure 1: Block diagram of the column-level architecture.

memory bank is composed of four capacitors which store the reset and output voltages of the pixel being encoded and its prediction. The pixel prediction circuit carries out the image decorrelation process. Due to area constraints the prediction circuit only uses two neighboring pixels to predict the current pixel: the pixel immediately above (N) and the pixel to the left (W). Although only two neighbors are considered in the prediction, compression rates comparable to more involved prediction techniques can be achieved [10]. The difference between the pixel being encoded X and its prediction $\hat{X}$ is computed by taking advantage of the differential input of the A/D.

The Predictor Selector circuit selects whether the neighboring pixel N or W or the value of the pixel in the previous frame will be selected as the pixel predictor. The prediction selection can be forced externally for testing purposes or it can be performed internally by the adaptive encoder. The internal predictor selector compares the result of the A/D conversion against a threshold. If the conversion is larger or equal than the threshold the predictor is changed to W or N. This algorithm is described in more detail in Section 2. A single-slope A/D is employed in this work because it can be directly integrated with a low-complexity entropy encoder by sharing common circuitry [11]. The read-out logic enables a serial read out of the variable-length codeword. Each column-level processor occupies an area of 3252 μm $\times$ 34.30 μm in a 4-metal, 2-poly 0.35μm CMOS technology.

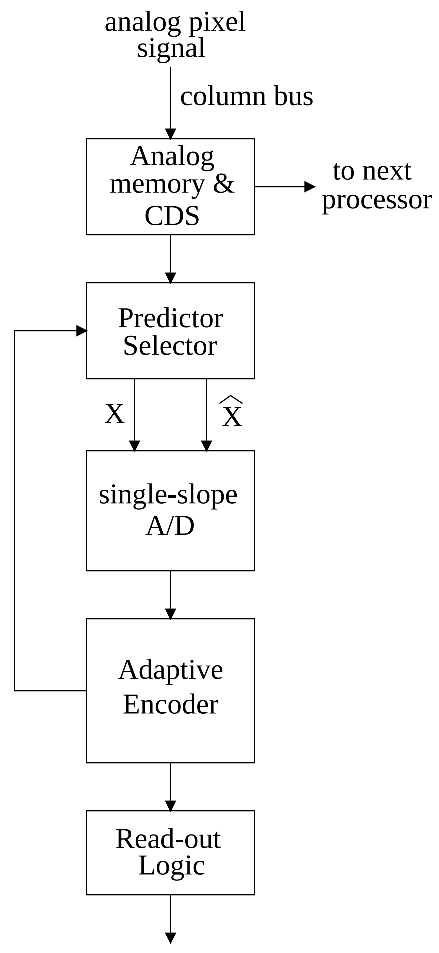


Figure 2: Block diagram of a column processor.

Analog Prediction vs. Digital Prediction

The pixel prediction is carried out in the analog domain. The main motivation to perform the prediction in the analog domain is that it can be implemented with smaller circuits. Another advantage of analog circuits is that the device physical principles can be used to implement complex mathematical equations. Image processing tasks such as filtering [12], [13], motion detection [15]-[17], color segmentation [18], optical flow measurement [14], cellular neural networks (CNN) [20], [21], and feature extraction [19] have been implemented on the focal plane with analog circuits. A drawback of analog circuits is their sensitivity to noise and to fabrication processes variations. Dynamic range in analog implementations is limited by supply voltage and noise levels. Typically the dynamic range in analog circuits is in the order of 40 to 100dB [9].

On the other hand, digital circuits have the advantage of achieving higher dynamic range. A 32-bit word can have a dynamic range of more than 800dB using floating point representation. They are less sensitive to noise and to technology scaling. However, digital circuits occupy much larger silicon area.

Next, an analysis on the impact of noise in an analog pixel predictor is carried out. The results are compared to an all-digital pixel prediction circuit. Figure 3 depicts the signal flow diagram of the analog pixel predictor.

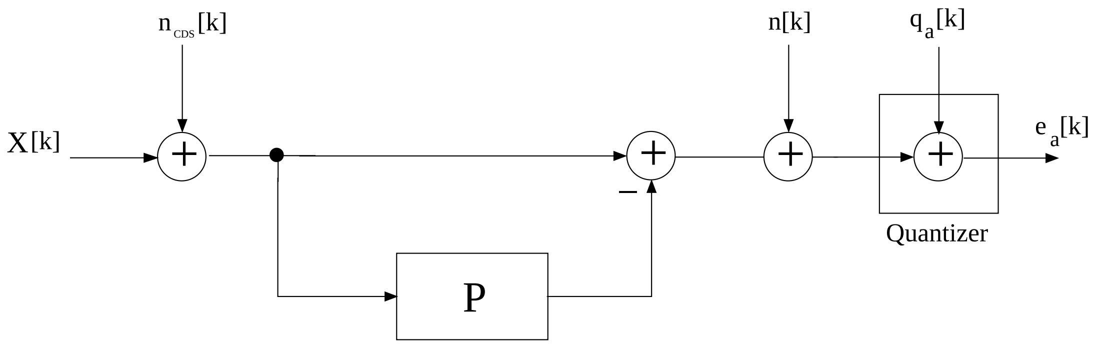


Figure 3: Signal flow diagram of the analog pixel predictor.

In the diagram, $X[k]$ represents the analog value of the pixel being encoded. In this notation the index k represents the spatial location of the pixels. In a raster scan only one pixel is processed at any given time, thus, the index k can also be viewed as a time index.

The Quantizer block models the response of the A/D. The quantizer is viewed as a source of additive white noise [24]. The input $q_a[k]$ denotes the quantization noise introduced by the quantizer and it is assumed to have zero mean and to be uniformly distributed. The quantizer is assumed to be a uniform mid rise N-bit quantizer with a quantization step $\Delta = V_{ref}/2^N$, where V_{ref} is the magnitude of the input range. The analog noise introduced by the predictor is modeled by the input $n[k]$ and the input $n_{\text{CDS}}[k]$ models the electronic noise introduced by the CDS circuit. Both $n_{\text{CDS}}[k]$ and $n[k]$ are assumed to be zero mean random variables. Notice that the quantizer is placed at the end after the prediction value and the prediction error have been computed. This set up has the advantage of allowing the prediction error to be computed with more compact analog circuits.

The response of the predictor block P is given by

$$P(X[k]) = \sum_{i=1}^{L} c_i \; X[k-i] \tag{1}$$

where c_i $(i = 1 \cdots L)$ are the prediction coefficients of an L^{th} order predictor. From Figure 3 the prediction error e_a can be written as:

$$e_a[k] = e[k] + n[k] + q_a[k] + n_{CDS}[k] - \sum_{i=1}^{L} c_i \cdot n_{CDS}[k-i] \tag{2}$$

where $e[k]$ is the prediction error without quantization noise, sometimes called uncontaminated error [23] and it is given by

$$e[k] = X[k] - \sum_{i=1}^{L} c_i X[k-i] \tag{3}$$

and $n_{CDS}[k]$ is the noise introduced by the CDS circuit.

The mean-square value of the prediction error e_a can be readily obtained from (2):

$$E\{e_a^2\} = \sigma_{e_a}^2 = {\sigma_e}^2 + {\sigma_n}^2 + {\sigma_{q_a}}^2 + \sigma_{CDS}^2 + \sigma_{CDS}^2 \sum_{i=1}^{L} c_i^2 + 2E\{e \cdot q_a\} \tag{4}$$

where it was assumed that e, n_{CDS}, and n are uncorrelated random processes, and that e_a is a zero mean random variable.

Figure 4 shows the block diagram of the digital pixel predictor. As in the analog case, the quantizer is a uniform mid rise quantizer. From Figure 4:

$$\tilde{e}[k] = X[k] - \sum_{i=1}^{L} c_i \; \bar{X}[k-i] + n_{CDS}[k] \tag{5}$$

but

$$\bar{X}[k] = X[k] + q_d[k] + n_{CDS}[k] \tag{6}$$

Combining equations (5) and (6)

$$\tilde{e}[k] = X[k] - \sum_{i=1}^{L} c_i (X[k-i] + q_d[k-i] + n_{CDS}[k-i]) + n_{CDS}[k] \tag{7}$$

From equations (7) and (3)

$$\tilde{e}[k] = e[k] - \sum_{i=1}^{L} c_i q_d[k-i] - \sum_{i=1}^{L} c_i n_{CDS}[k-i] + n_{CDS}[k] \tag{8}$$

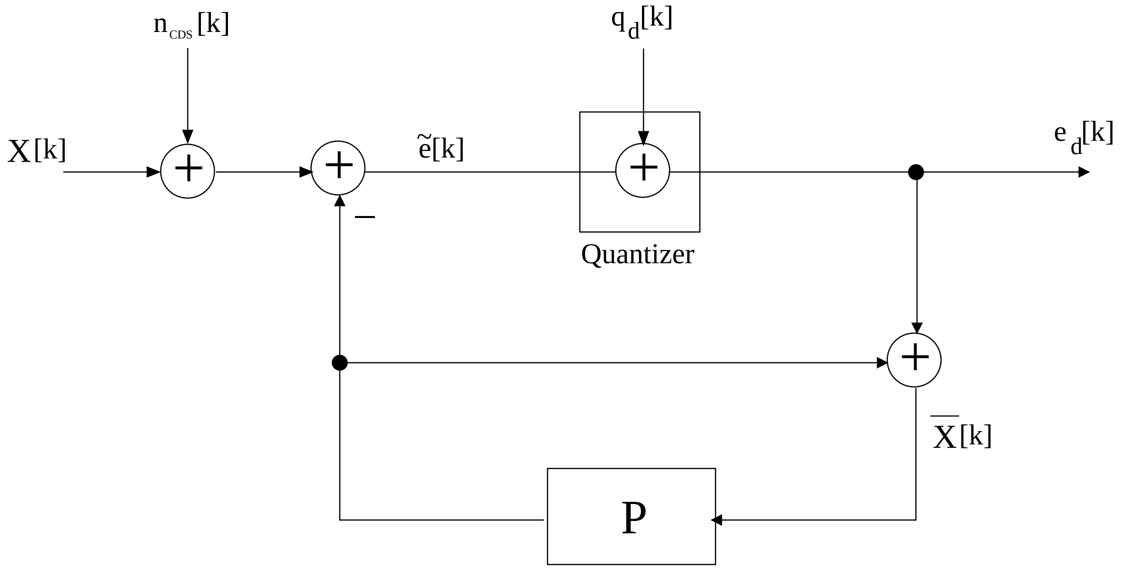


Figure 4: Block diagram of the digital pixel predictor.

Assuming that $\tilde{e}$ is a zero mean random variable and that the quantization noise q_d is uncorrelated with itself, with the uncontaminated prediction error e, and with the CDS circuit noise, the mean-square value of the error $\tilde{e}$ turns out to be

$$E\{\tilde{e}^2\} = \sigma_{\tilde{e}}^2 = \sigma_e^2 + (\sigma_{q_d}^2 + \sigma_{CDS}^2) \sum_{i=1}^{L} c_i^2 + \sigma_{CDS}^2 \qquad (9)$$

From Figure 4

$$e_d[k] = \tilde{e}[k] + q_d[k] \qquad (10)$$

Thus, the mean square value of e_d is

$$E\{e_d^2\} = \sigma_{e_d}^2 = \sigma_{\tilde{e}}^2 + \sigma_{q_d}^2 + 2E\{\tilde{e} \cdot q_d\} \qquad (11)$$

where it was assumed that e_d is a zero mean random variable. Replacing (9) into (11)

$$\sigma_{e_d}^2 = \sigma_e^2 + \sigma_{q_d}^2 + (\sigma_{q_d}^2 + \sigma_{CDS}^2) \sum_{i=1}^{L} c_i^2 + \sigma_{CDS}^2 + 2E\{\tilde{e} \cdot q_d\} \qquad (12)$$

Subtracting equation (12) from (equation 4),

$$\sigma_{e_a}^2 - \sigma_{e_d}^2 = \sigma_n^2 + \sigma_{q_a}^2 + 2E\{e \cdot q_a\} - \sigma_{q_d}^2 - \sigma_{q_d}^2 \sum_{i=1}^{L} c_i^2 - 2E\{\tilde{e} \cdot q_d\} \qquad (13)$$

To ensure that the noise performance of the analog-based predictor is better or equal than the noise performance of the digital predictor, the following condition must be met:

$$\sigma_{e_a}^2 - \sigma_{e_d}^2 \leq 0 \tag{14}$$

Since the quantizers for the analog and digital cases are uniform with the same number of quantization steps, it follows that $\sigma_{q_a}^2$ equals $\sigma_{q_d}^2$ and the inequality (14) is reduced to

$$\sigma_n^2 \leq \sigma_{q_d}^2 \sum_{i=1}^{L} c_i^2 - 2E\{e \cdot q_a\} + 2E\{\tilde{e} \cdot q_d\} \tag{15}$$

To compute the correlation terms $E\{e \cdot q_a\}$ and $E\{\tilde{e} \cdot q_d\}$, we will assume that e has a Laplacian distribution. Laplacian distributions are often employed to model prediction residuals [22]. The correlation between the input and the quantization noise of a quantizer can be readily evaluated with the following equation derived in [25].

$$E\{XN\} = -\{\sigma_x^2 - 2 \sum_{i=1}^{2^{N-1}} Y_i^2 P[x_{i-1} \leq X < x_i]\} \tag{16}$$

where X is the quantizer input, N is the quantization noise, and Y_i are the output levels of the quantizer.

Also notice from equation (8) that $\tilde{e}$ is the sum of Laplacian and uniform distributed random variables. Its probability density function (pdf) is the convolution of a Laplacian and a uniform pdfs. Further simplifications can be done by noticing that the range of the quantization noise q_d, Δ, for the current implementation is small ($\Delta = V_{ref}/2^N = 3.3/256 = 0.0129$). It follows that the pdf of $\tilde{e}$ approximates a Laplacian distribution with zero mean and variance σ_e^2. Thus, the inequality (15) is reduced to

$$\sigma_n^2 \leq \sigma_{q_d}^2 \sum_{i=1}^{L} c_i^2 \tag{17}$$

To evaluate $\sigma_{q_d}^2$ recall that for a uniform quantizer the variance of the quantization noise is $\sigma^2 = \frac{\Delta^2}{12}$ [22]. For the current application which targets a 0.35μm CMOS technology with a 3.3 V supply, and considering a quantizer with $N = 8$ bits of resolution, the quantization step Δ turns out to be 12.9 mV. Thus, the variance $\sigma_{q_d}^2$ equals 1.3847×10^{-5} V^2. In the present design only two neighbor pixels and the previous frame value are employed in the prediction, resulting in $L = 3$. Furthermore, the predictor selector circuit selects between the neighbors W and N and the previous frame value in a sequential manner. Therefore, only one of the coefficients c_i is one at any given time and the rest are zero. This implies that $\sum_{i=1}^{L} c_i^2$ equals 1. Replacing these values in (17), we obtain an upper bound on the noise introduced by the analog pixel prediction block, i.e. $\sigma_n^2 \leq 13.847 \ \ \mu V^2$. This bound is compared to the noise introduced by the analog prediction circuit in Section 2.

In this work we implement an analog predictor due to its area and noise advantages. However, when performing the prediction in the analog domain, the quantization error gets accumulated in the decoding process. To illustrate this point consider the decoder shown in Figure 5 and assume that at the beginning of the encoding process the prediction is known without error to the decoder, i.e. $\hat{X}_r[1] = \hat{X}[1]$. From Figure 5 the decoded pixel value X_r is

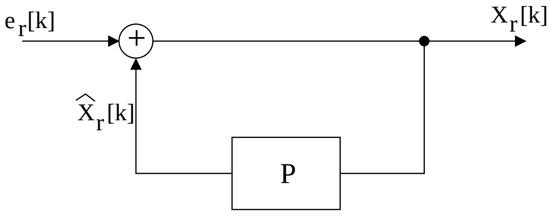


Figure 5: Block diagram of the decoder.

$$\begin{aligned} X_r[1] &= \hat{X}_r[1] + e_a[1] \\ &= X[1] + e_r[1] \end{aligned} \tag{18}$$

where $e_r[k] = n[k] + q_a[k] + n_{CDS}[k] - \sum_{i=1}^{L} c_i \cdot n_{CDS}[k-i]$ (see equation (2)). Let us assume without loss of generality that the pixel predictor simply takes the previous decoded pixel as the new prediction, thus, $\hat{X}_r[k+1] = X_r[k]$. It follows that

$$\begin{aligned} X_r[2] &= \hat{X}_r[2] + e_a[2] \\ &= X_r[1] + e_a[2] \\ &= X[2] + e_r[1] + e_r[2] \end{aligned} \tag{19}$$

In general

$$\begin{aligned} X_r[k] &= \hat{X}_r[k] + e_a[k] \\ &= X_r[k-1] + e_a[k] \\ &= X[k] + \sum_{m=1}^{k} e_r[m] \end{aligned} \tag{20}$$

From the last equation the quantization error will accumulate without bound in the decoder. To prevent this accumulation the timing signals of the A/D have to be properly chosen. This issue will be dealt with in more detail in the next section.

Circuit Design

A prototype chip has been designed and fabricated in a 0.35μm CMOS technology. The prototype chip occupies a silicon area of 2597.50μm $\times$ 5958.90μm. The chip contains an APS array of 80 $\times$ 44 pixels, 44 column-level processors, vertical and horizontal shift register for sequential addressing of rows and columns, and a 44-to-1 multiplexer for readout. There are two vertical shift registers. The first one controls the reset transistors in one row and the second one enables one row at a time for readout. This way a roll-down reset operation of the rows can be performed. In a roll-down scheme the reset and the readout signals are delayed by the integration time. This allows a faster readout of pixels without having to wait a pixel integration period in each row.

After a pixel row is enabled for readout and the reset and photo-generated voltages are sampled, the analog-to-digital conversion is started. Due to the integration nature of the A/D, 128 clock cycles are required for conversion. In the non-compressing mode 9 bits are read out for each column, of which 7 are significant bits and one is a sign bit. In the compressing mode the number of bits read out is variable. It can vary from 2 bits to 21 bits. With a clock frequency of 50 MHz, a video sequence of 100 frames/second can be obtained.

Pixel Circuit Design

APS pixels are employed in this design. The pixels include amplifiers that buffer internal signals and allow non-destructive reading. The pixel design should allow the following functionality: read-out of the photo-integrated voltage, the reset voltage, and the photo-integrated voltage in the previous frame, and the storage of the value that the pixel had on the last frame. To accomplish this last function, a poly1/poly2 capacitor has been embedded into the pixel design.

Figure 6 shows the schematic diagram of the APS pixel. In the schematic, the capacitor C_m serves to store the previous frame pixel voltage. The transistors M2 and M3 form a source follower amplifier that allows writing into the capacitor C_m without changing the photodiode voltage. Transistors M4 and M5 work as switches controlling the write and read-out operations. Transistors M6 and Mb form a second source follower amplifier whose purpose is to drive the bus load. The transistor Mb is shared by all the pixels in the same column. The PMOS transistor M1 is the reset transistor. By using a PMOS transistor for the reset operation, the voltage range of the photodiode is maximized. Finally, transistors M7 and M8 form a transmission gate that connects the pixel to a column-shared bus. The operation of the pixel has

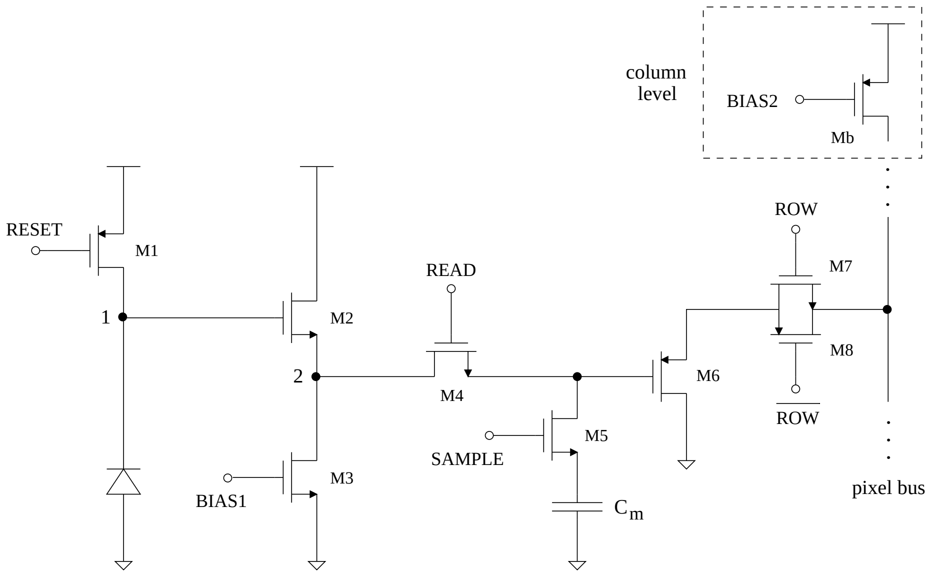


Figure 6: Schematic diagram of a pixel.

three phases: reset, integration, and read-out. Figure 7 shows a timing diagram of the pixel control signals during these phases.

In the reset phase transistor M1 is turned on resetting the photodiode to VDD volts. In the integration phase transistor M1 is turned off causing the photo-generated current to discharge the photodiode junction capacitance. The read-out phase starts when the transistor M5 is turned on while M4 remains off. At the same time the transmission gate formed by transistors M7 and M8 is turned on. Thus, the voltage across capacitor C_m can be read out through the PMOS source follower composed by M6 and Mb. After this first readout transistor M4 is closed. At this point the output of the M2-M3 source follower is stored in the capacitor C_m and it is also read out trough the M6-Mb buffer. The last voltage to be read out from the pixel is the reset voltage. To do this the switch M5 opened and M1 is closed. The pixel is reset and the reset voltage can be read out. The read-out cycle finishes when when all switches are opened.

Figure 8 depicts some of the simulated internal waveforms of the pixel. Notice the offset of the voltages at nodes 1 and 2 due to the gate-to-source drop of transistor M2. Given that the sizes of transistors M2 and M3 are the same and without considering second order effects, the offset voltage between nodes 1 and 2 is close to the bias

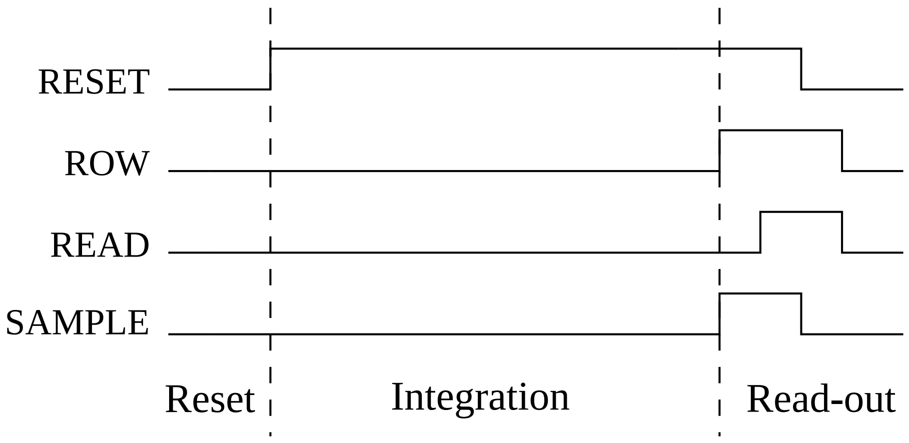


Figure 7: Pixel timing diagram.

voltage BIAS1 which in this case is 1 V. This offset voltage is compensated to a degree by the M6-Mb PMOS source follower. Nevertheless, the gate-to-source drop of transistor M2 limits the pixel dynamic range. The pixel layout occupies an area of 34 μm $\times$ 30 μm in a CMOS 0.35μm technology and has a fill factor of 18%. The photodiode is made of an n-well on top of a p-substrate. The capacitor C_m has a capacitance value of 150 fF. Figure 9 shows the layout of a pixel.

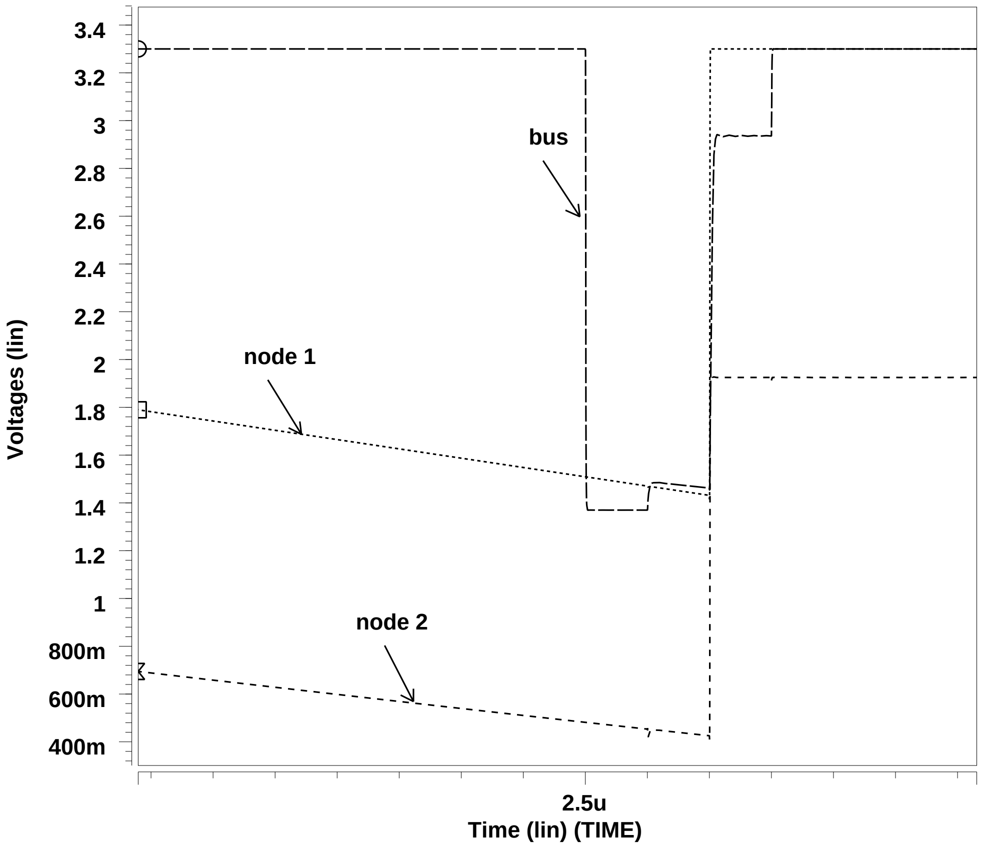


Figure 8: Pixel waveforms.

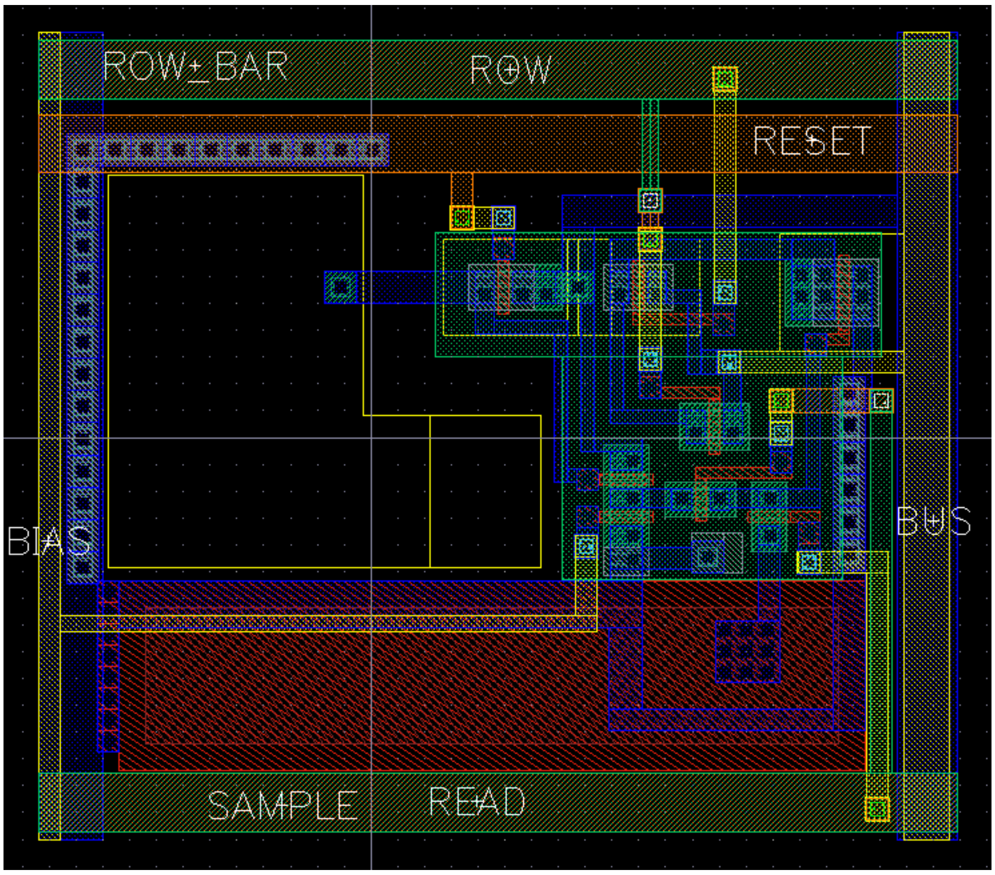


Figure 9: Layout of the pixel.

Prediction Circuit

The pixel prediction can be one of two modes: intra-frame or inter-frame. In the intra-frame mode the pixels W and N are employed as predictors. In the inter-frame mode the pixel value one frame before, $X(t - t_f)$, is considered as the pixel prediction. The function of the pixel prediction circuit is to provide to the A/D the

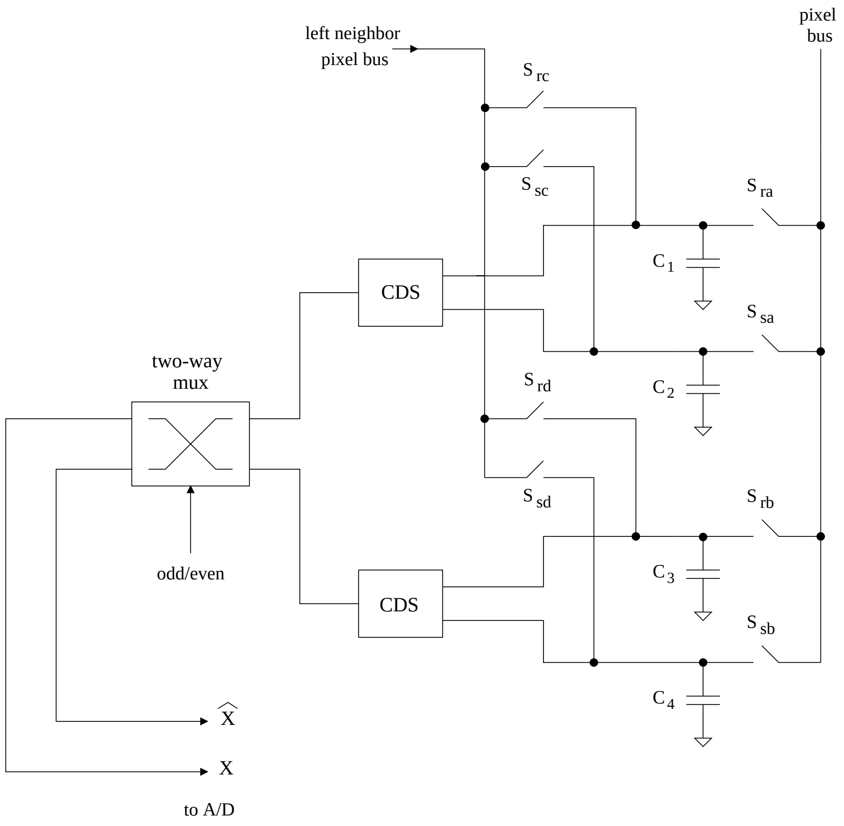


Figure 10: Block diagram of the pixel predictor circuit.

values of the pixel being encoded, X, and its predicted value, $\hat{X}$ in the right order. The selection of the pixel prediction depends on the state of the signals **intra/inter**, **W/N**, and **odd/even**. The **odd/even** line alternates between **1** and **0** every time the row number changes. If the active row number is an odd number, **odd/even** is set to **1**, otherwise it is reset to **0**.

Figure 10 shows the diagram of the pixel prediction circuit. The capacitors C_1 to C_4 work as a memory bank to store the voltages from the pixel X and its prediction

$\hat{X}$. The storage of pixel value X is alternated between the capacitor pairs $C_1 - C_2$ or $C_3 - C_4$. The **odd/even** line keeps track of which capacitor pair currently stores the pixel value X. The other capacitor pair will be storing the value of the prediction $\hat{X}$. The capacitors C_1 and C_3 store the reset voltages while C_2 and C_4 store the voltages after integration of the corresponding pixels X and $\hat{X}$. The reason we alternate the storage between the capacitor pairs is to be able to sample a new row of pixels while keeping the row sampled before, which after sampling the new row becomes the previous row. The previous row contains the N pixel employed in the prediction process, thus, it is necessary to store it. The function of the two-way multiplexer is to guarantee that the values of X and $\hat{X}$ are presented to the A/D in the right order.

Figure 11 shows the schematic diagram of the multiplexer. When the multiplexer's input **dir**, which is connected to the **odd/even** line, is **1** the output **x** gets connected to the input **a** and **y** to **b**, otherwise **x** gets connected to **b** and **y** to **a**. Each

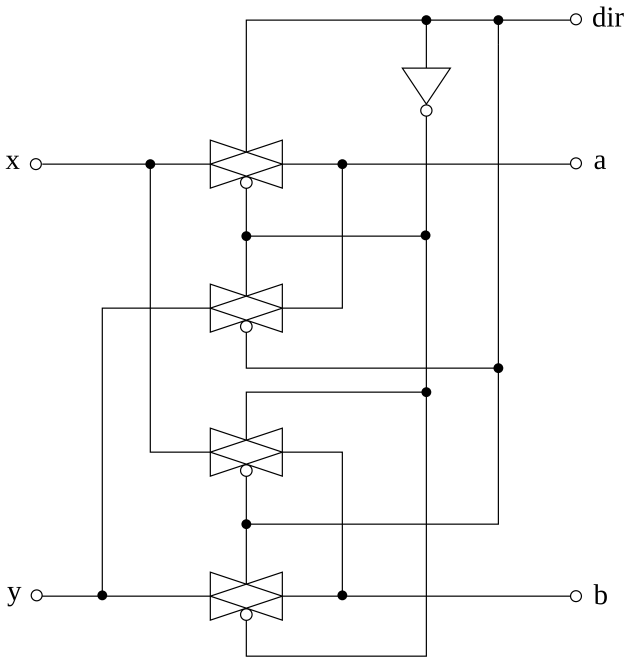


Figure 11: Schematic diagram of a the two-way mux.

transmission gate in the multiplexer is composed of four transistors as shown in Figure 12. The function of the transistors M3 and M4 is to minimize the charge injection impact from transistors M1 and M2.

The pixel sampling process is illustrated in Figure 13. The signal **intra/inter** determines whether an intra-frame or an inter-frame prediction will be carried out. When **intra/inter** $=$ **0** the prediction value is set to the pixel value one frame

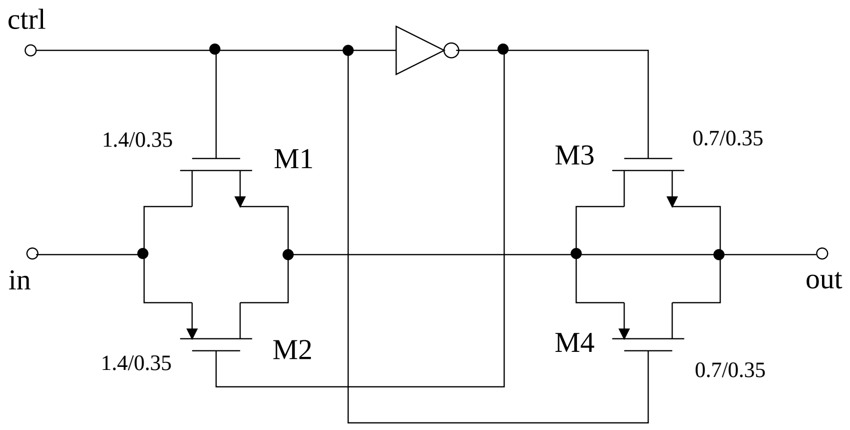


Figure 12: Schematic diagram of the transmission gate.

before, i.e. $\hat{X} = X(t - t_f)$, where t_f is the frame period. Otherwise, if **W/N**=1, the prediction value will be set to the value of the pixel W and if **W/N**=0, it is set equal to the value of pixel N. When an odd row is being read-out the reset voltages of X and $\hat{X}$ are sampled in C_1 and C_3 respectively, and the photo-generated voltages are sampled in C_2 and C_4. Similarly, when an even row is read-out, C_2 and C_4 store the reset voltages and C_1 and C_3 the photo-generated voltages.

The signals **intra/inter** and **W/N** can be controlled externally or they can be generated internally by the Predictor Selector circuitry. The Predictor Selector circuit compares the output of the A/D with a threshold input THR. If the output of the A/D is greater than the threshold, the predictor is changed according to the algorithm shown in Figure 14. Since the output of the A/D is the quantized version of the prediction error, the algorithm is essentially comparing the prediction error and the threshold value.

The signal **video/still** is used to force a prediction mode. If **video/still**=0, the prediction is always in the intra frame mode. Otherwise, the algorithm switches back and forth from the intra-frame to the inter-frame modes.

The schematic diagram of the Predictor Selector circuit is shown in Figure 15. To simplify the hardware design a 3-bit threshold is employed. Even with three bits the threshold has enough range to allow for programmability.

The digital circuit of Figure 16 controls the switches S_{ra}, S_{rb}, S_{rc}, S_{rd}, S_{sa}, S_{sb}, S_{sc}, and S_{sd}. In the figure the inputs S_d, S_s, and S_r are the sampling signals for the photo-generated the reset voltages. Their timing in relation with the RESET, READ, and SAMPLE control signals is shown in Figure 17.

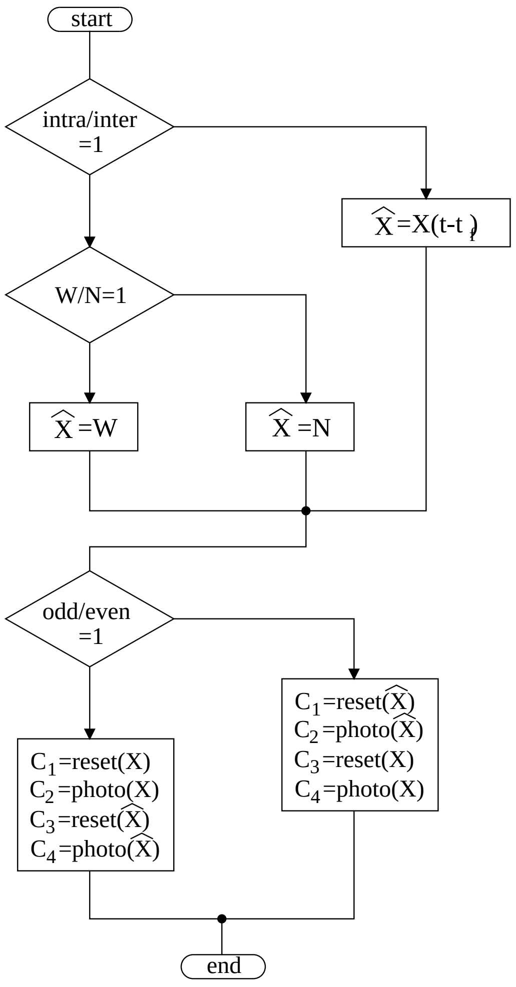


Figure 13: Flow diagram of the pixel selection process.

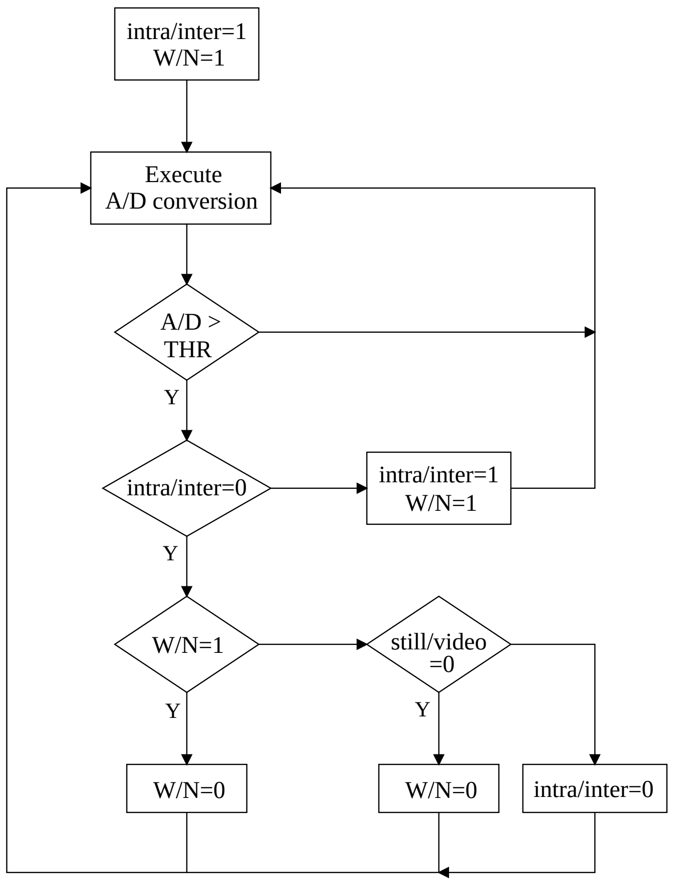


Figure 14: Flow diagram of the predictor selection process.

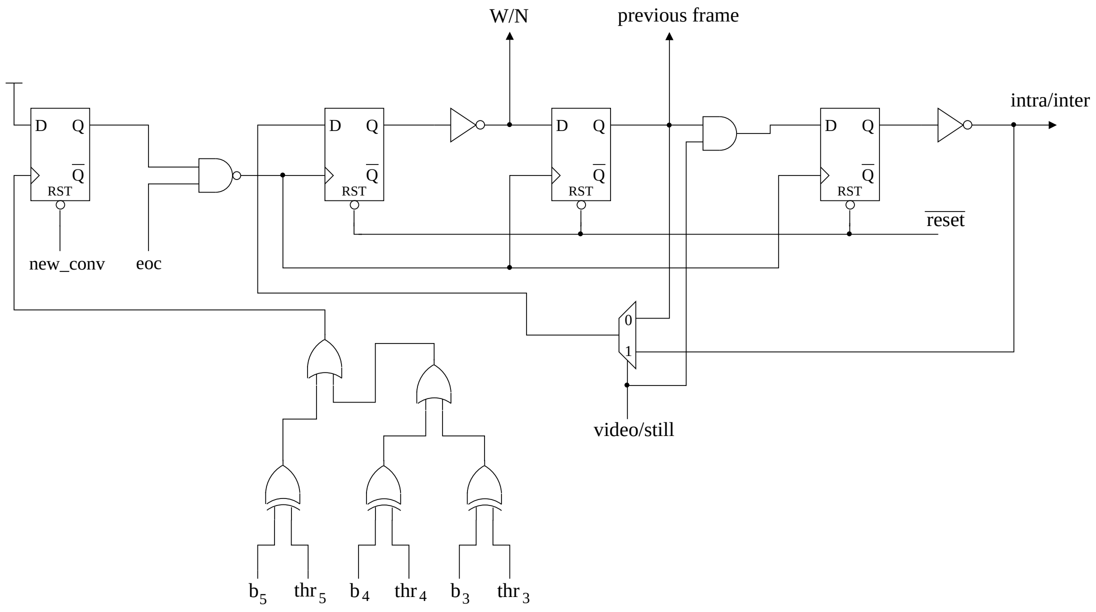


Figure 15: Schematic diagram of the predictor selector circuit.

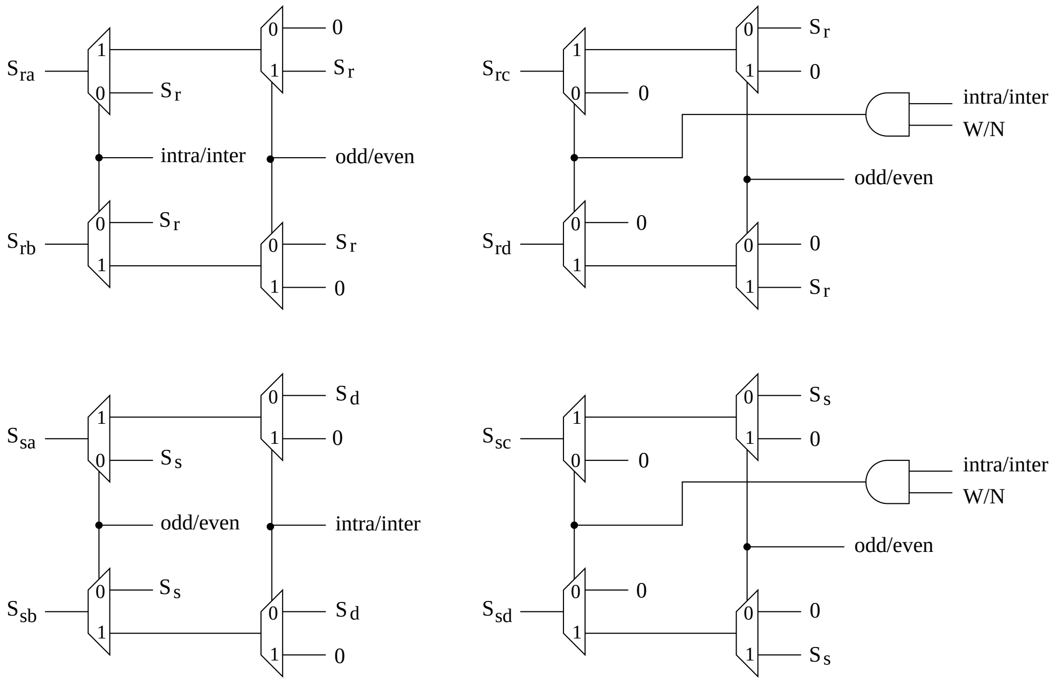


Figure 16: Schematic diagram of the switch control circuits.

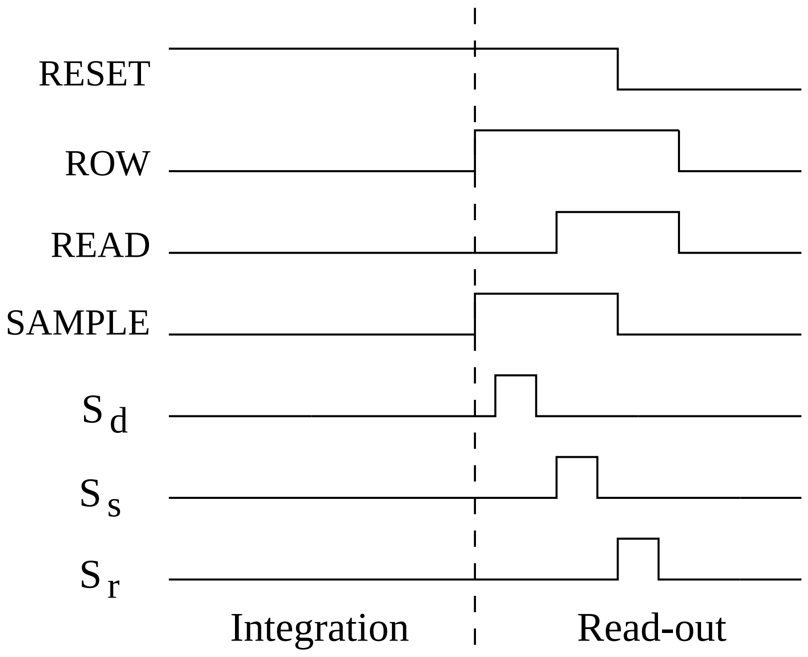


Figure 17: Diagram of the column-level Predictor Selector circuit.

Prediction Circuit Noise Analysis

A noise analysis of the prediction circuit of Figure 10 will be conducted to determine the impact of the analog noise in the prediction performance. Specifically, we would like to compare the noise power with the bound of equation (17). From Figure 10 notice that the noise introduced by the sampling switches $S_{sa,ra}$, $S_{sb,rb}$, $S_{sc,rc}$, and $S_{sd,rd}$ and the comparator will affect both the analog-based and the digital-based prediction circuits as they have to be present in both cases. The CDS circuit noise is already accounted for in the analysis of Section **??** where it was shown that this noise affects both the analog and the digital predictors. Thus, the 2-way multiplexer is the only block contributing to the noise $n[k]$.

Each transmission gate is composed of four transistors as shown in Figure 12. Transistors M3 and M4 are half the size of transistors M1 and M2 and their function is to mitigate the charge injection effects [32].

The noise contribution from these MOS switches is computed by considering their ON resistance R_{on} as a source of white noise with spectral density given by

$$V^2_{ron}(f) = 2kTR_{on} \tag{21}$$

The noise bandwidth is limited by the gate input capacitance of the A/D comparators (see Figure 21). The final rms noise generated by one transistor can be computed by considering the white noise as being filtered by a first-order, low-pass RC filter. In mathematical terms,

$$V_{ron}^2(\text{rms}) = \int_0^\infty \frac{2kTR_{on}}{1+\left(\frac{f}{f_0}\right)^2}\,df = 2kTR_{on}\pi\left(\frac{1}{2\pi R_{on}C_{gs}}\right) = \frac{kT}{C_{gs}} \tag{22}$$

where k is Boltzman's constant, T is the absolute temperature in K, and C is the gate input capacitance. We assume a room temperature of 300K. The input capacitance is given by [28]

$$C_{gs} = \frac{2}{3}WLC_{ox} \tag{23}$$

where $W = 1.2\mu$m and $L = 0.4\mu$m and $C_{ox} = 4.5 \times 10^{-15}F/\mu m^2$ is the oxide capacitance. Replacing these values in (23) yields $C_{gs} = 1.44$fF. The rms noise generated by one transistor is then $V_{ron-rms}^2 = 1.437\mu V^2$. Since there are eight transistors contributing to the noise, the total noise is $11.4960\mu V^2$. Comparing this value with the upper bound of equation (17) we conclude that in terms of noise it is better to perform the prediction in the analog domain. Certainly there is also a complexity advantage of an analog implementation over a digital one.

Correlated Double Sampling

The CDS circuit is a two-transistor differential amplifier [33], [34]. Its schematic diagram is shown in Figure 18. Neglecting second order effects, the output of the circuit is given by

$$V_{out} = V_{DD} - (V_r - V_s) \tag{24}$$

For correct operation both transistors M1 and M2 must remain in the saturation region. The transistors will work in the saturation region if the following inequalities are satisfied

$$\begin{aligned} 2V_r - V_s &> V_{DD} - V_{th} \\ V_{DD} - V_{th} &> V_r \end{aligned} \tag{25}$$

where V_{th} is the transistor threshold voltage. In the schematic diagram the input V_r represents the pixel reset voltage and the input V_s the photo-generated pixel voltage. When there is no incident light the voltage V_s equals the voltage V_r and the ideal

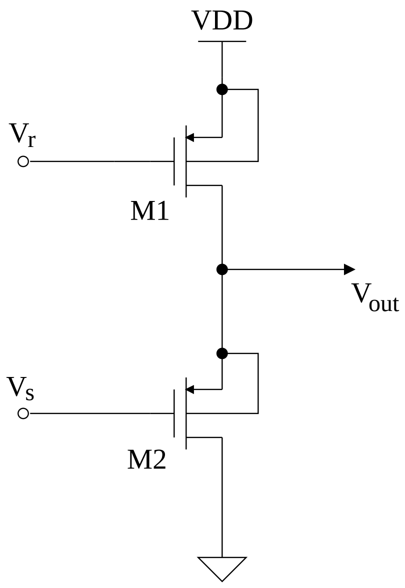


Figure 18: Schematic diagram of the correlated-double-sampling circuit.

output of the circuit is V_{DD}. However, this will take transistor M1 out of the saturation region. To avoid this problem two different reset voltages will be employed. The first reset voltage V_{r1} is employed during the actual reset of the pixel and the second reset voltage V_{r2} is employed during the read-out operation. Let $V_{r2} = V_{DD} - V_{th} - \Delta$ with $\Delta > 0$, condition (25) is satisfied if $V_{r1} < V_{DD} - V_{th} - 2\Delta$. In practice Δ should be large enough to absorb the variations of V_{th} from column to column but at the same time should be kept small so as to maximize the dynamic range of V_s. It has been found through simulations and experimentation, that, for the targeted CMOS process, where $V_{DD} = 3.3$ V and V_{th} is around 0.7 V, the optimal values for V_{r1} and V_{r2} are 2.4 V and 2.5 V respectively. The values of these voltages are changed by changing the voltage BIAS2 of the column-level transistor Mb.

To minimize second order effects such as short channel, the sizes of transistors M1 and M2 were chosen to be W=4.8μm and L=1.2μm . The layout of the CDS circuit pair has been carefully designed to minimize transistor mismatches. Figure 19 shows the layout of a pair of CDS circuits. The circuits occupy an area of 23.60μm $\times$ 34.30μm. Both CDS circuits have been placed in close proximity to minimize the impact of the fabrication process variations. Active areas have been shielded from light with metal 4.

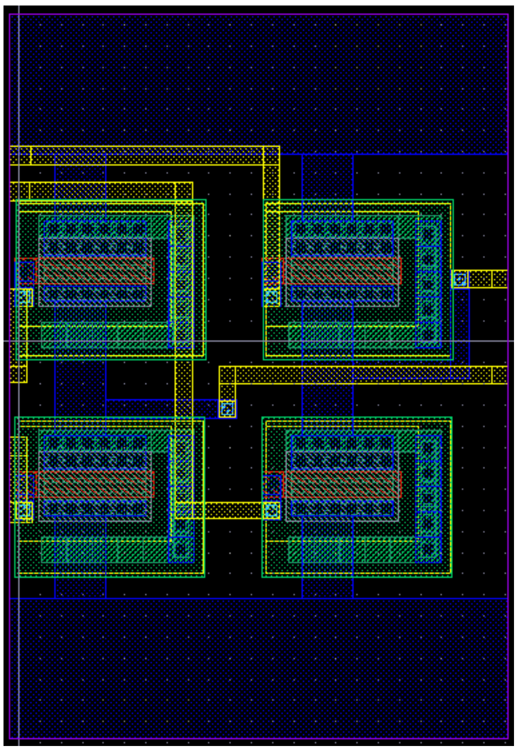


Figure 19: Layout of the CDS circuit.

Figure 20 shows the simulation of the CDS circuit. It can be seen that circuit is very linear and its output closely follows the ideal output. This high linearity, however, comes at the price of power consumption. The CDS circuit consumes 18 μW.

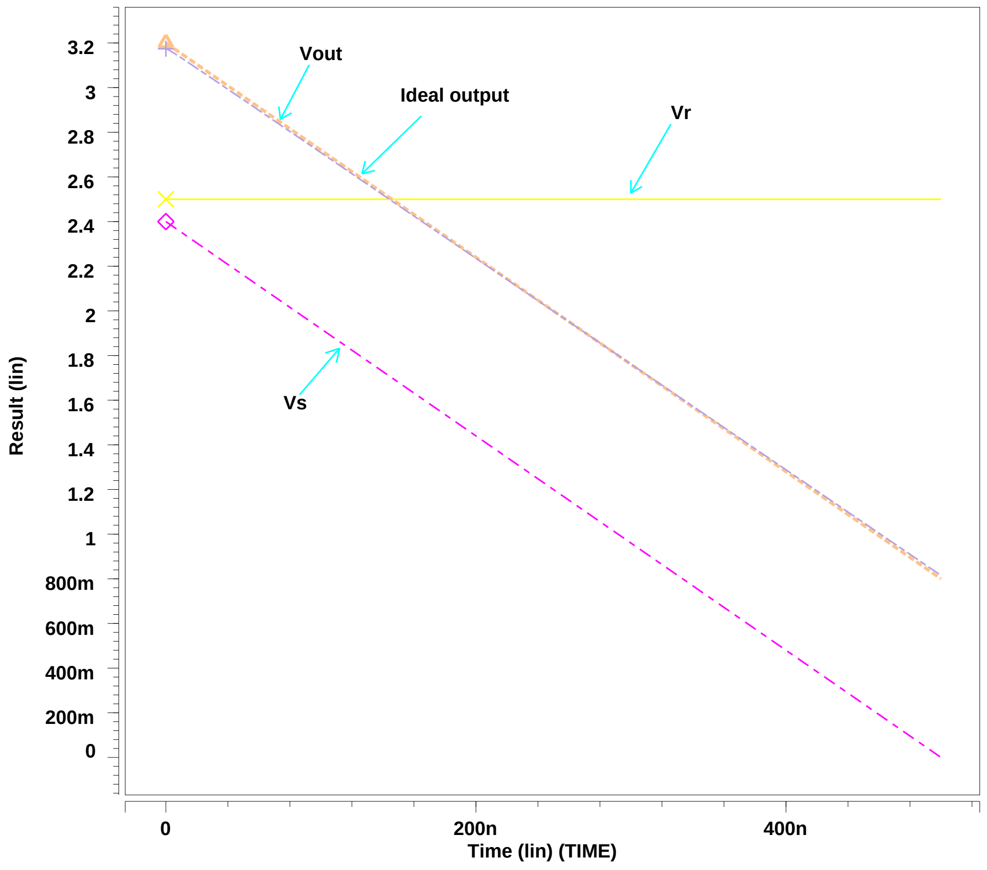


Figure 20: CDS circuit waveforms.

Adaptive Encoder Circuit

The Adaptive Encoder block is composed of a single-slope A/D, an adaptive Golomb-Rice encoder, and read-out logic that allows a serial read-out of the code-words. A schematic diagram is shown in Figure 21.

The value of current pixel X and the predicted value $\hat{X}$ are compared with a reference voltage ramp V_{ref}. The schematic diagram of the comparators is shown in Figure 22. The comparator is composed by a differential amplifier followed by a

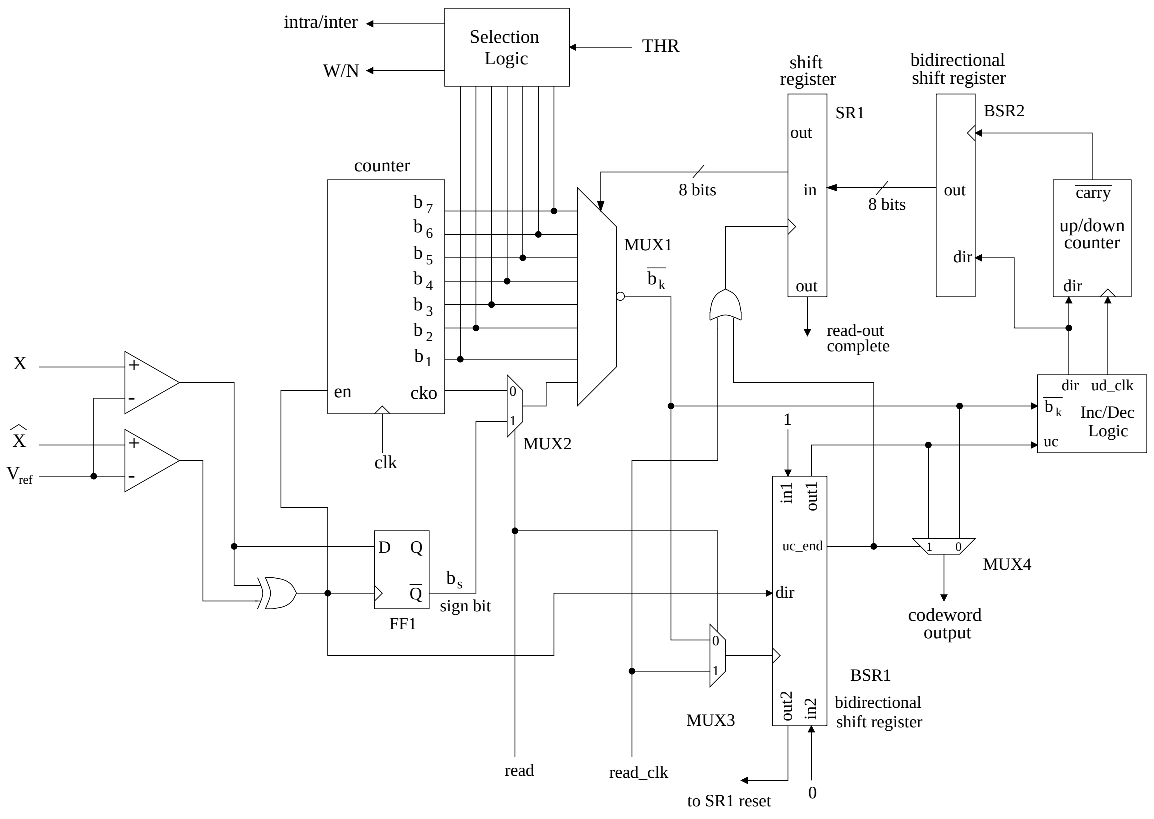


Figure 21: Schematic diagram of A/D and the entropy encoder.

common-source amplifier formed by the transistors M6 and M7. To minimize power consumption these amplifiers are operated in the sub threshold region. The total power consumption of the comparator is 642.223 nW with a gain of 214. The output is buffered by two cascaded inverters that drive the downstream load. The offset voltage of the comparator is 12.9 mV. Since both comparators are affected by the same offset, their impact on the analog-to-digital conversion is minimized as they will cancel each other out.

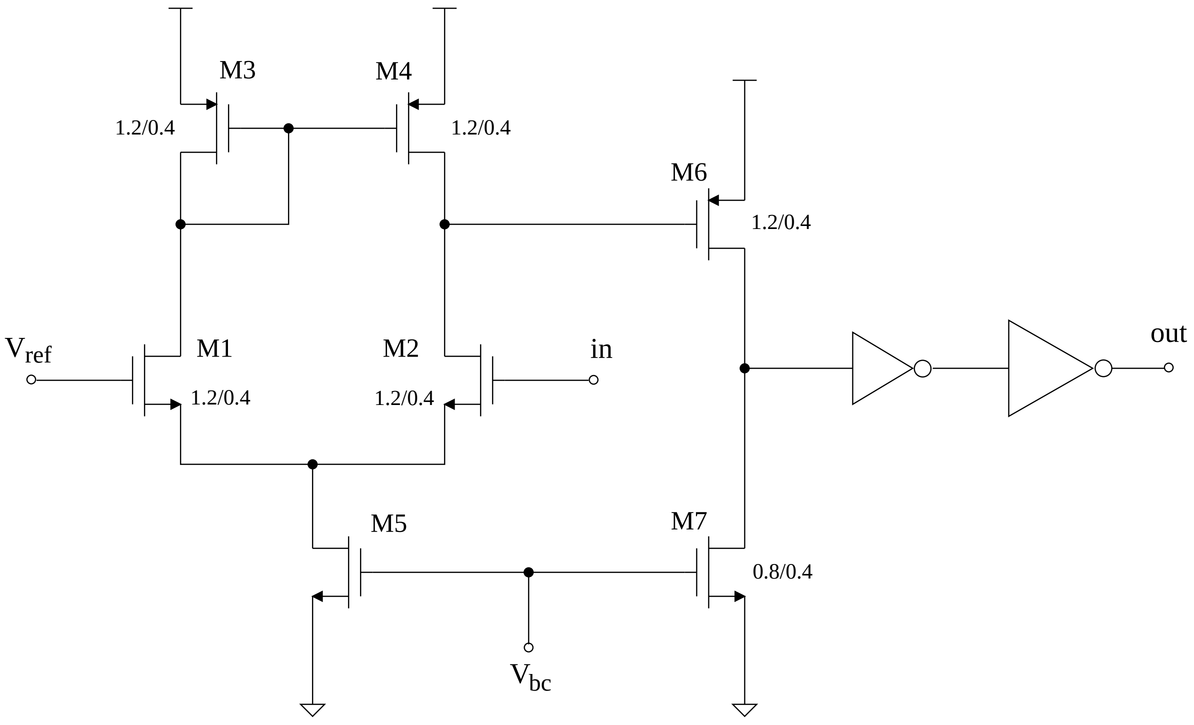


Figure 22: Schematic diagram of the voltage comparator.

Figure 23 shows the layout of a comparator pair. The layout occupies an area of 39.05 μm $\times$ 34.30 μm. The width of the layout matches the pixel pitch. The active areas have been covered with metal 4 to shield them from light. All the transistors have been placed in close proximity to minimize the impact of fabrication process variations.

The analog-to-digital conversion is carried out by comparing the voltages representing X and $\hat{X}$ with the voltage ramp V_{ref} and detecting the crossing points t_1 and t_2 (see Figure 24). With this operation the prediction error, $X - \hat{X}$, is transformed from the voltage domain into the time domain. Once in the time domain the error can be quantized with a digital counter. The **clk** signal in Figure 24 drives

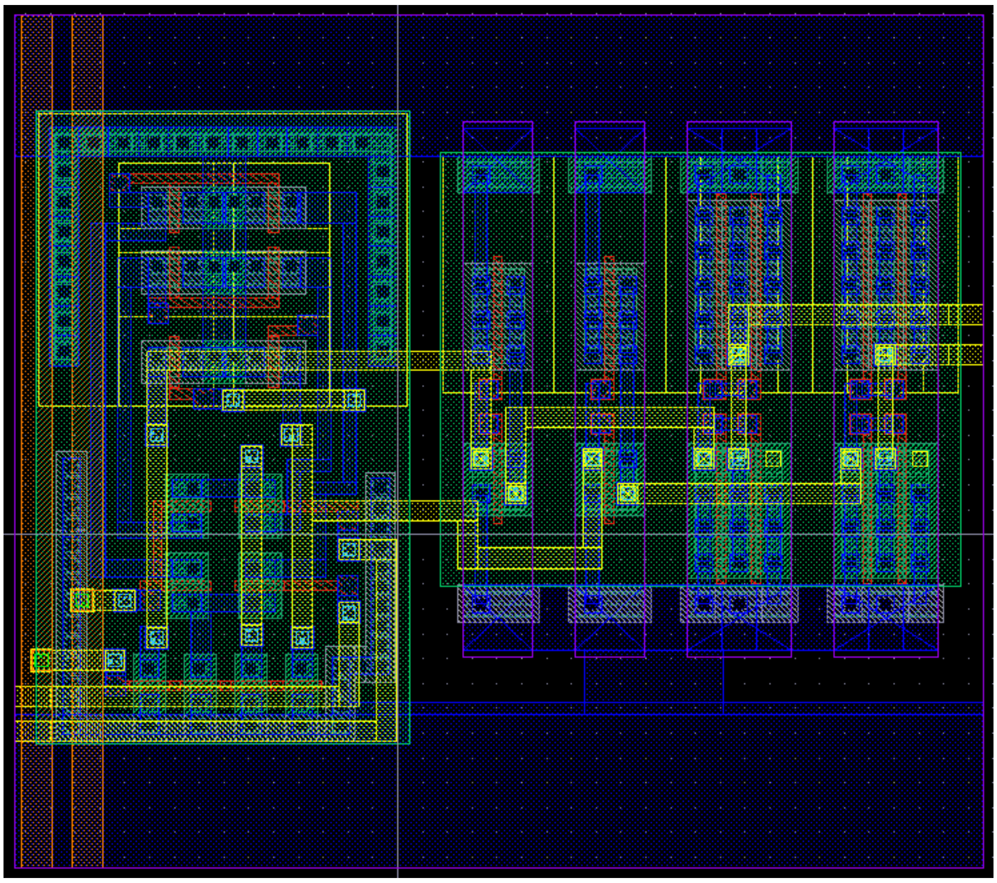


Figure 23: Layout of the voltage comparator.

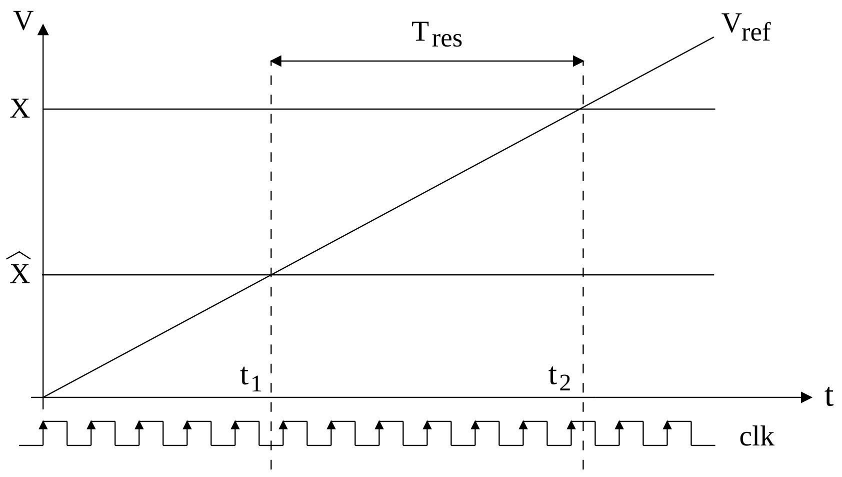


Figure 24: Time domain waveforms for a single slope conversion.

the counter's clock input. A simple change in the timing of the clock signal can avoid the accumulation of the quantization error in the decoder. To this end the clock rising edge should coincide with the start of the voltage ramp V_{ref} as shown in Figure 24. The counter employed is a 7-bit synchronous up-counter. Since the counter can only count positive numbers, a sign bit is needed to completely specify the prediction residual. The sign bit is generated by the XOR gate and the flip-flop FF1. The sign bit is set to **1** when $X \geq \hat{X}$ and to **0** when $X < \hat{X}$. The conversion phase requires 128 clock cycles. Despite their longer conversion time, single-slope integrating A/Ds have been integrated in CMOS imagers due to their simplicity and linearity advantages. Although faster A/D architectures could have been employed, a single-slope integrating A/D provides a unique opportunity for the integration of an A/D with an entropy encoder [11]. The entropy encoder employed in this work is a Golomb-Rice encoder. This type of encoder is known for its complexity advantages. Despite its simple structure it can achieve compression ratios similar to more complex architectures like the Huffman or the arithmetic encoder [10]. The digital counter of the single-slope A/D can be used to generate Golomb-Rice codes with a given coding parameter k. The unary code is generated by counting how many times the counter overflows the counting range given by the bits $b_1, \cdots, b_k$. This condition is detected when the bit b_k goes from **1** to **0**. The binary part of the codeword is just the counter least significant bits $b_1, \cdots, b_k$. The diagram of the 7-bit digital counter is shown in Figure 25.

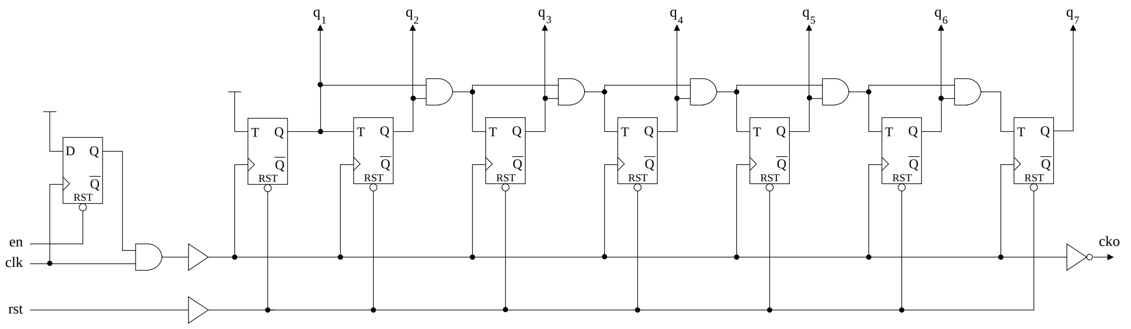


Figure 25: Schematic diagram of the digital counter.

An 8-to-1 multiplexer is employed to select the k^{th} bit from the counter. The output of the multiplexer drives the clock input of the bidirectional shift register BSR1 through MUX3. Every time b_k goes from **1** to **0** a **1** is shifted in. Thus, at the end of the conversion BSR1 contains the unary code. A bidirectional register is employed to ease the read-out operation. During read-out BSR1 is shifted in the opposite direction and the unary code is available at its output **out1**. The register

BSR1 has a length 12 bits. If the unary code is longer than or equal to 12 bits, the shift register SR1 is reset causing a read out of all the counter bits plus the sign bit. In other words, there will be no compression with the 12-bit unary code acting as an escape symbol. The maximum codeword length occurs in this situation and is equal to 12+1+8=21 bits. Figure 26 shows the schematic diagram of the register BSR1.

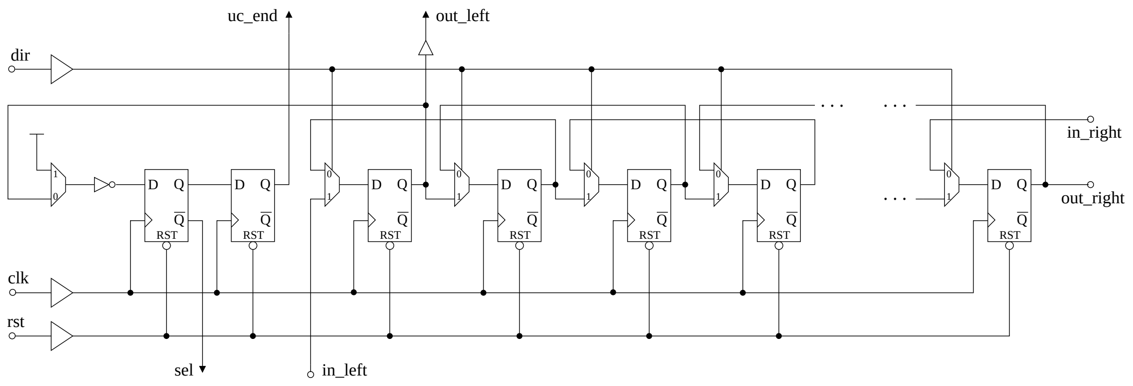


Figure 26: Schematic diagram of the bidirectional shift register.

The Golomb-Rice codes are effective if the parameter k is properly tuned to the statistics of the input data. Otherwise, compression gains would be marginal or even worse, expansion would occur. The low-complexity adaptive algorithm is analyzed in detail in Section 5 has been implemented and integrated at the column level next to the A/Ds. The adaptive algorithm is implemented by the Inc/Dec Logic and the up/down counter. Figure 27 shows the schematic diagram of the Inc/Dec Logic circuit. The flip-flop FF1 detects the condition **8**: `If` $(u_k = 0$ `AND` $b_{k-1} = 0)$ while flip-flop FF2 detects conditions **4** to **7**. The up/down counter block implements the conditions **9** and **10**. u_k is the length of the unary code of the current codeword and b_{k-1} is the k^{th} bit of the binary counter. The timing of the control inputs **eoc**, **new_conv**, and **clk2** are depicted in Figure 28. This pattern is repeated in each analog-to-digital conversion.

Figure 29 shows the schematic diagram of the 3-bit up/down counter. Three bits suffice to count the range $-M$ to M when M is 4. M is a threshold input to the coder and remains constant throughout the image encoding. When the control input **dir** is **1** the counter counts up and when **dir** equals **0** it counts down. The up/down counter is reset to the value $q_3, q_2, q_1 = 100$. Whenever it reaches the value 000 it resets itself and produces a negative pulse on its carry output. This pulse will drive the clock input of the bidirectional shift register BSR2. The flip-flop FF4 prevents the up/down counter from counting after it reset itself and before a new conversion cycle starts. At any point in time the contents of BSR2 are all **0** except for one position

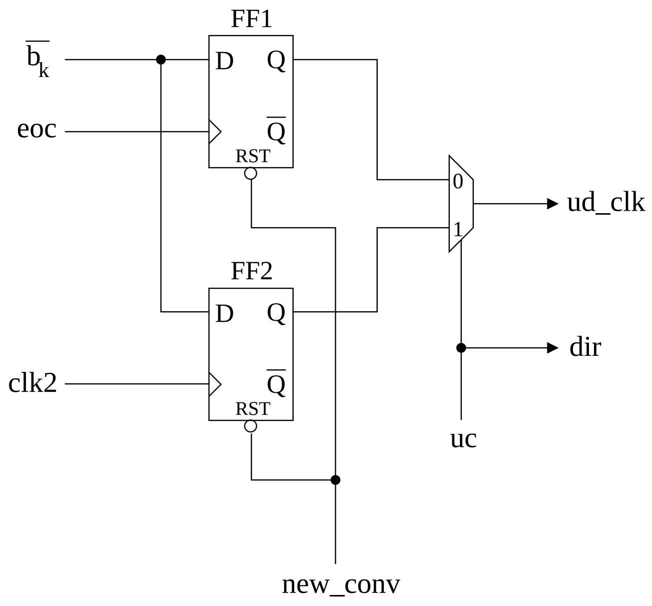


Figure 27: Schematic diagram of the condition detection logic.

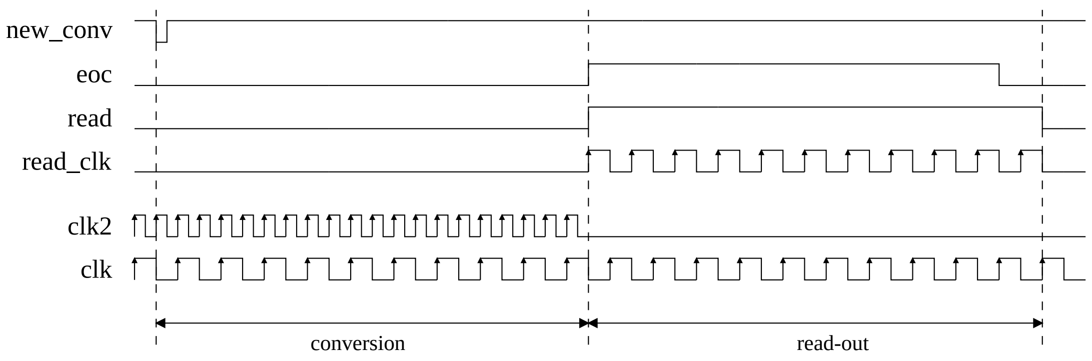


Figure 28: Timing diagram of the encoder control signals.

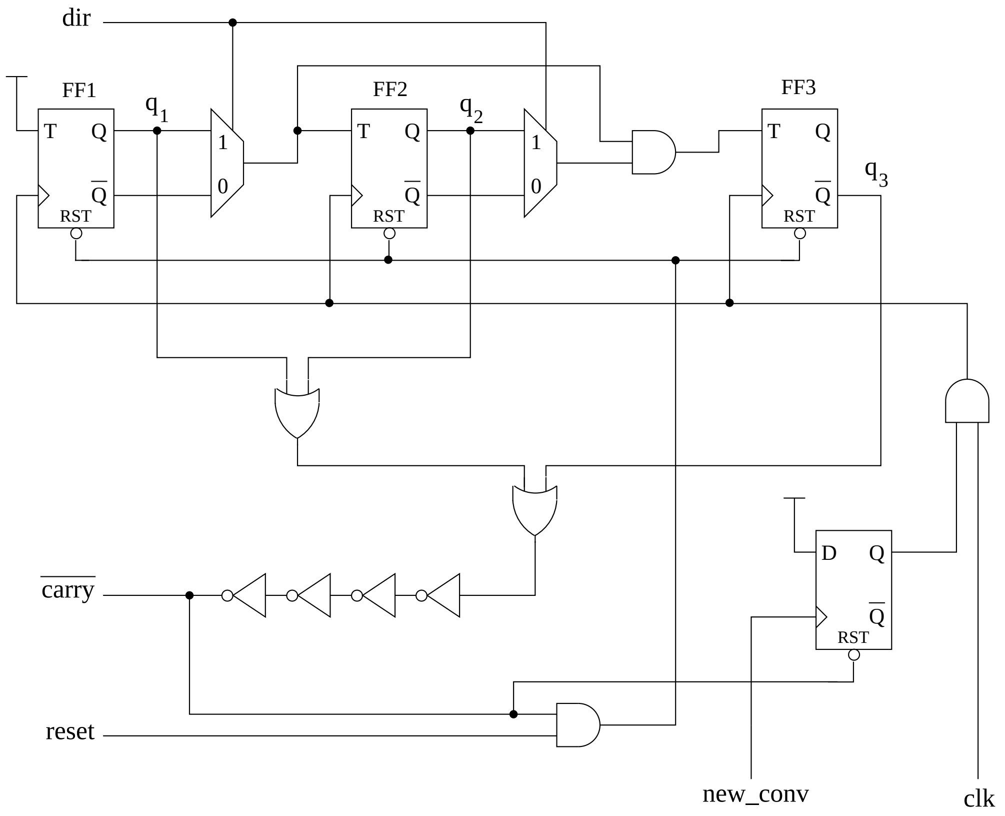


Figure 29: Schematic diagram of the 3-bit up/down counter.

which is a **1**. The value of k is encoded in the position of the **1** bit. After reset the most significant bit of BSR2 is set to **1** and the rest of the bits are set to **0**. When the up/down counter overflows, the contents of register BSR2 are shifted to the right, thus, incrementing the value of k. Similarly, when the up/down counter underflows, the contents of BSR2 are shifted to the left, decrementing the value of k. Circuits at both ends of BSR2 prevent the **1** from leaving the register.

The read-out phase begins when the control signal **read** goes from **0** to **1**. During this phase the clock input of BSR1 is driven by the read-out clock **read_clk**. The unary code is shifted out the register BSR1 through its output **out1**. When all the unary code has been shifted out the output **uc_end** of the register BSR1 goes low enabling the read out of the second part of the codeword. The second part is composed of the counter bits $b_k, \cdots, b_1$ and the sign bit b_s. The shift register SR1 is employed during this operation. At the beginning of every conversion cycle the contents of BSR2 are loaded in parallel into SR1. Then the contents of SR1 are shifted out causing the register to successively select the counter bits b_k to b_1 and b_s through the multiplexer MUX1. Due to the inverting output of MUX1, these bits will be inverted. Each column processor occupies an area of 3252μm x 34.30μm in a 0.35μm CMOS technology. A more detailed block diagram is shown in Figure 30. The chip layout is shown in Figure 31. It occupies an area of 2596.50μm $\times$ 5958.90μm.

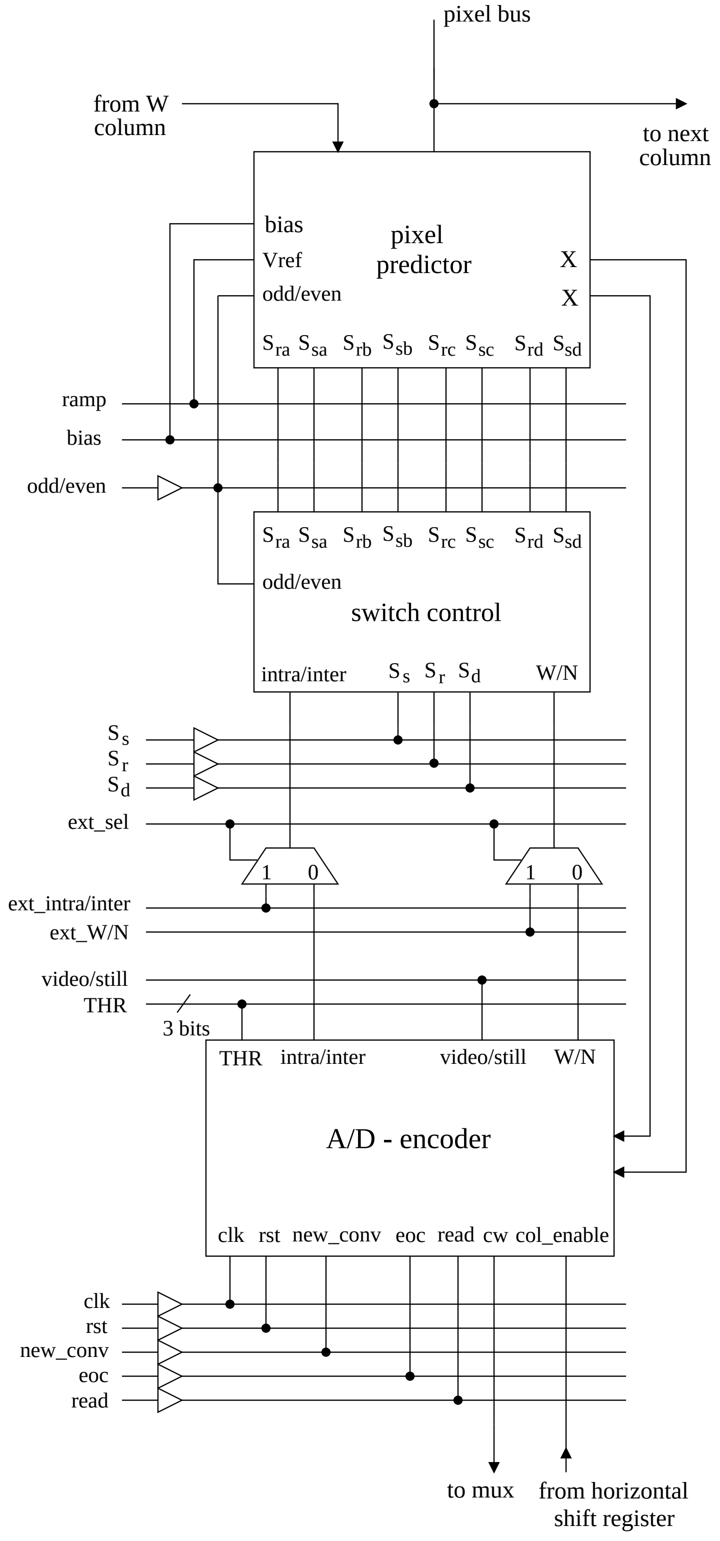


Figure 30: Block diagram of a column-level processor

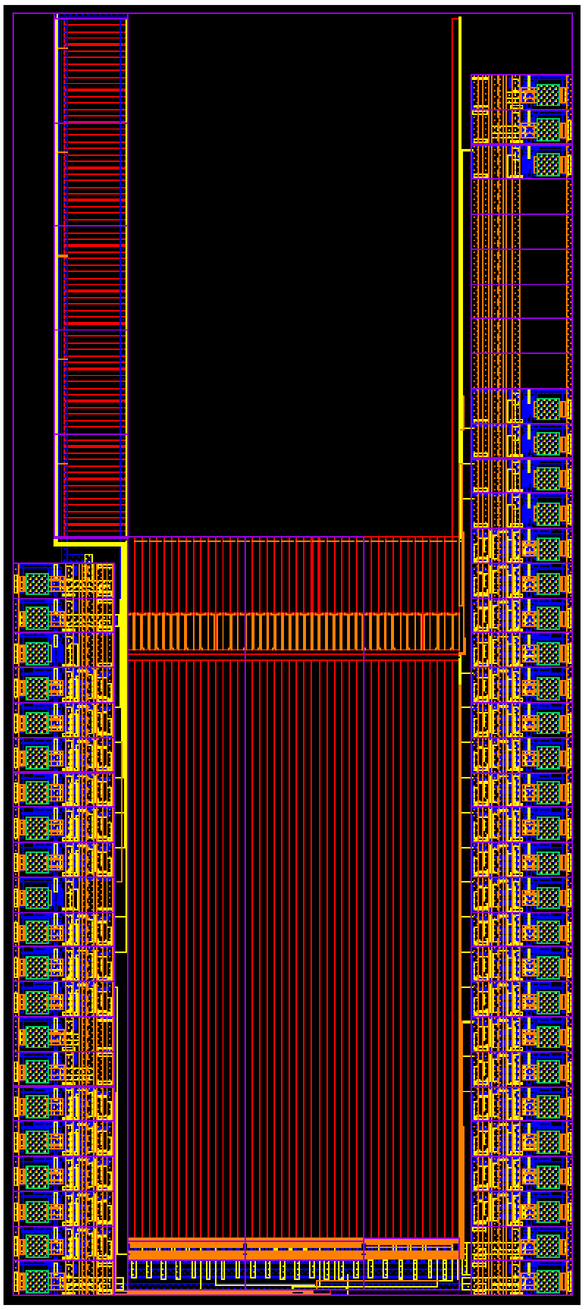


Figure 31: Layout of the imager chip.

3 Test Results

Test Set-Up

Figure 32 shows a diagram of the test set-up. The aim was to develop a camera with a USB interface to a Personal Computer (PC). The camera contains an ATMEL microcontroller, memory to store images, digital-to-analog converters (DACs) to generate bias voltages and a reference voltage ramp, and a USB interface to communicate with a PC. These components were placed together on a Printed Circuit Board (PCB) and mounted in a metal enclosure outfitted with a 3.5-8mm focal length, 1/3" format lens. Figure 33 shows the prototype camera and a close-up of the PCB.

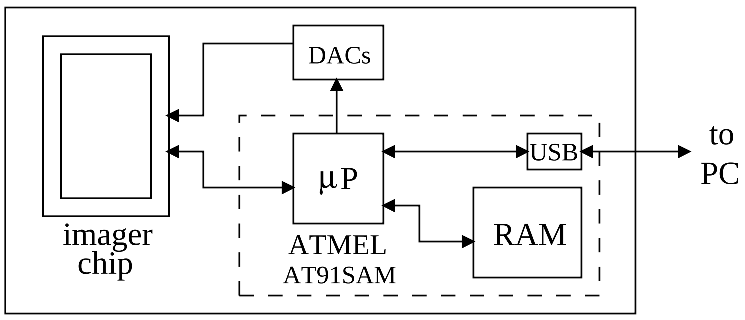
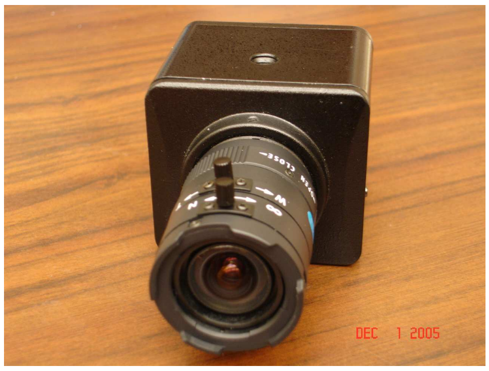


Figure 32: Test setup of the imager chip.

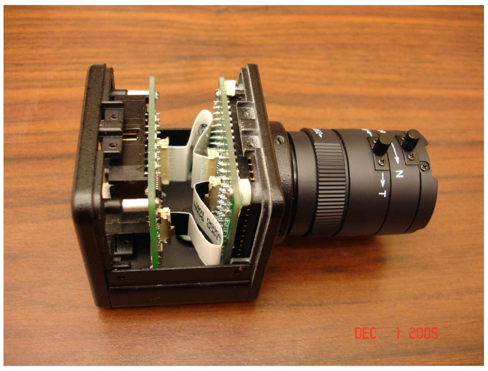
A/D Test

The chip contains an extra A/D whose input is tied to an access pad instead of a regular column output of the array. The A/D has eight bits of resolution. Seven of the Least Significant Bits (LSB) are provided by the digital counter and the Most Significant Bit (MSB) is the sign bit. First, the voltage input range of the A/D $V_{max} - V_{min}$, is established. V_{max} and V_{min} are the maximum and minimum input voltages that can be detected by the converter. From measurements, $V_{max} = 2.8$ V and $V_{min} = 1.0$ V. Thus, the input range is 1.8 V. This input range can be extended by employing rail-to-rail architectures in the voltage comparators but it is enough to accommodate the output of the PMOS column-level source followers. Thus, 1 LSB is equivalent to $V_{LSB} = 1.8/2^8$=7 mV.

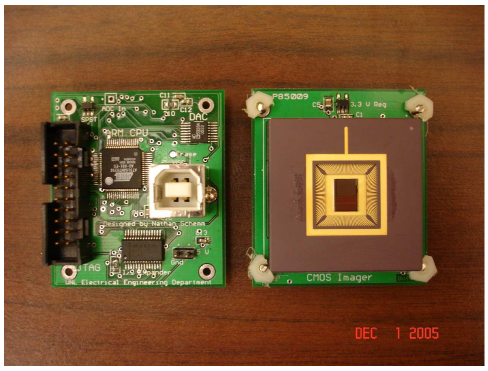
The A/D was tested using the code density technique described in [62], [63]. In that technique the A/D is tested at full speed and is based on randomly collecting



(a)



(b)



(c)

Figure 33: Camera: (a) front view; (b) inside view; (c) unmounted.

a large number of samples of a known periodic input and creating a histogram of the output of the A/D. An analysis of the histogram can reveal the Differential Non-Linearity (DNL), the Integral Non-Linearity (INL), the offset voltage, and any missing codes. The input of the A/D was chosen to be a voltage ramp which ideally should yield a uniform histogram. However, due to noise, offset and non-linearities in the converter, the measured histogram is not uniform. Figure 34 shows the measured histogram and the ideal or reference histogram. The large peaks around the output code 128 are due to inaccurate sign bit transitions. The inaccurate sign bit transitions are caused by sign bit detection circuit composed by the XOR gate and the flip-flop FF1 (see Fig. 21). Because the output of the XOR gate is used to sample the output of the top comparator, when $X > \hat{X}$ the rising edge in the flip-flop's clock will coincide with the transition of the top comparator. This causes the flip-flop to sample its input at the wrong time instant. To solve this problem a sufficient delay should be introduced in the XOR gate. The inaccurate sign bit transitions will also affect the DNL.

The gain of the A/D can be measured from the histograms with the following

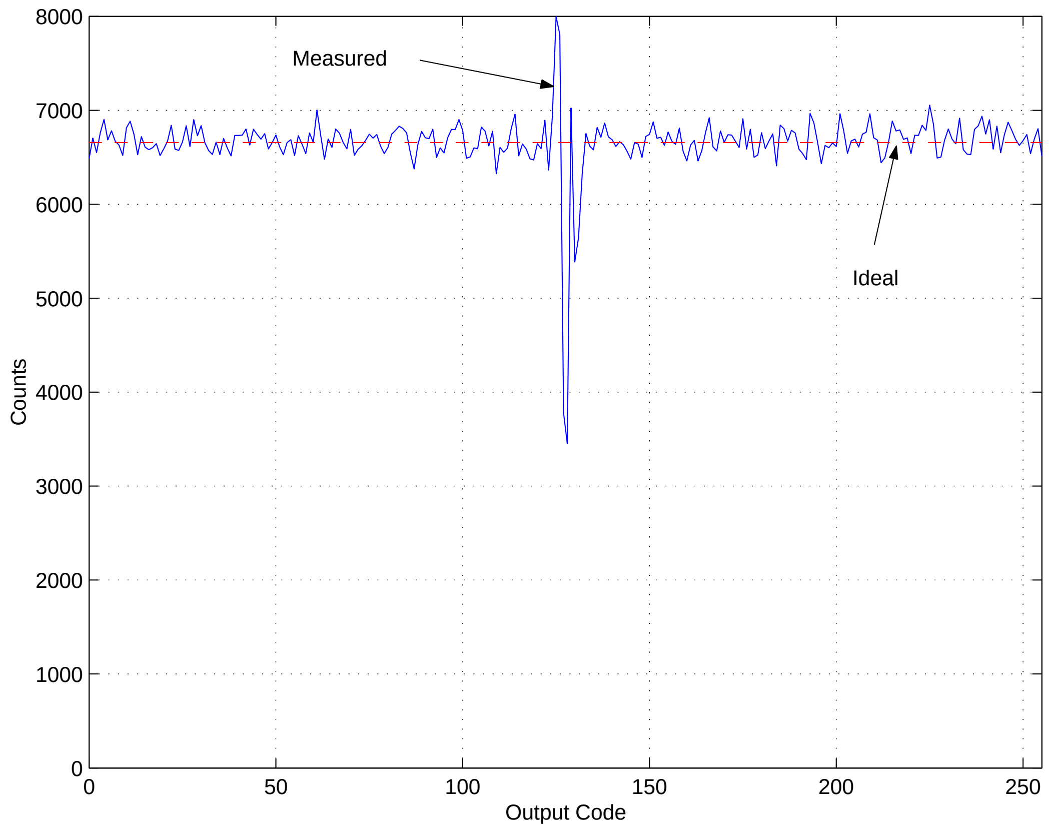


Figure 34: Histogram of the output of the A/D.

equation,

$$G = \frac{1}{2^N} \sum_{i=1}^{2^N} \frac{H_{ref}(i)}{H_{mes}(i)} \tag{26}$$

where $N = 8$ is the number of bits of resolution, H_{ref} is the reference or ideal histogram and H_{mes} is the measured or experimental histogram. The gain G turns out to be equal to 1.0039. The offset, V_o, is computed by

$$V_o = V_{LSB} \frac{\sum_{i=2^{N-1}+1}^{2^n} H_{mes} - \sum_{i=1}^{2^{N-1}} H_{mes}}{\sum_{i=1}^{2^N} H_{mes}(i)} \tag{27}$$

and it is equal to 2 mV.

The DNL which is defined as the variation from 1 LSB of the range of voltages that result in the same output code can be calculated from the measured histogram in the following way,

$$DNL(i) = \frac{H_{mes}(i) - H_{ref}(i)}{H_{ref}(i)} = \frac{H_{mes}(i)}{H_{ref}(i)} - 1. \tag{28}$$

Figure 35 shows the DNL. From the figure it is clear that the DNL is less than +/- 0.5

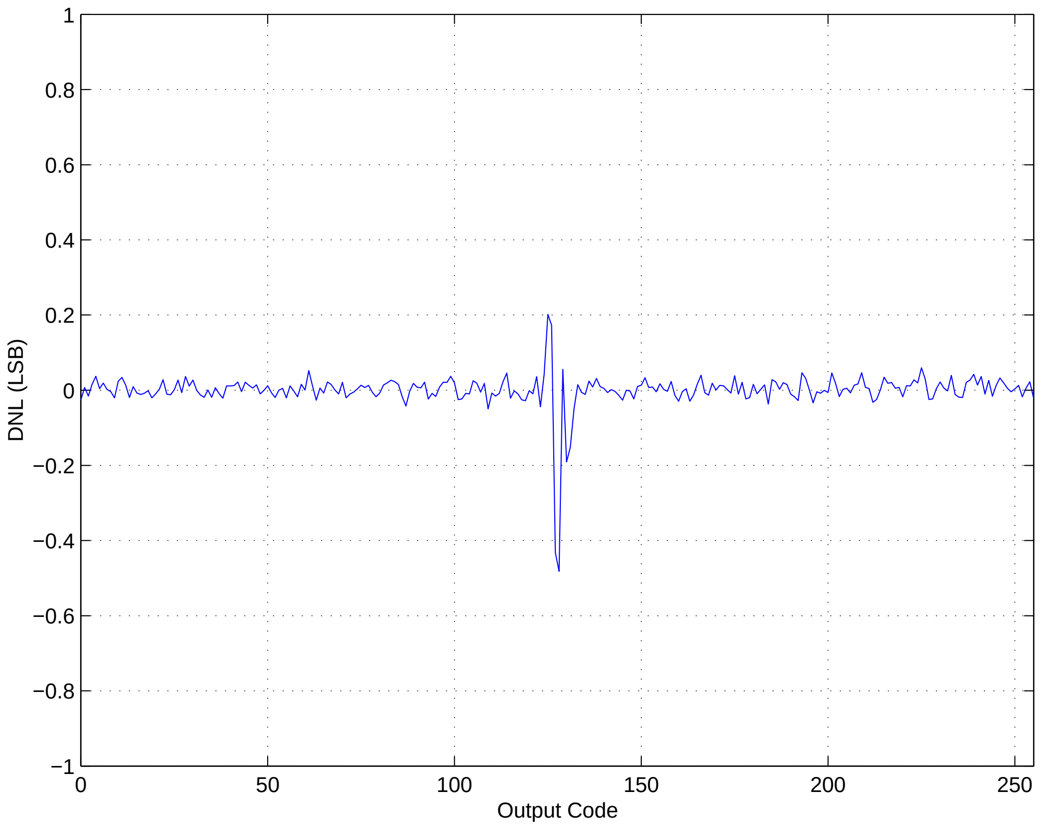


Figure 35: Differential non-linearity of the A/D.

LSB, therefore, the A/D has no missing codes. We arrive at the same conclusion by inspecting the measured histogram and noticing that none of the bins have a count of zero. The INL is defined as the deviation of the transfer curve from ideality or from a straight line [62], [28]. Experimentally it can be found as the cumulative sum of the DNL of all preceding codes [63], i.e.

$$INL(i) = \sum_{j=1}^{i} DNL(j) \tag{29}$$

Figure 36 shows the INL. The INL is less than +/- 0.6 LSB. The good linearity obtained is a result of the integrating architecture employed in the design of the A/D.

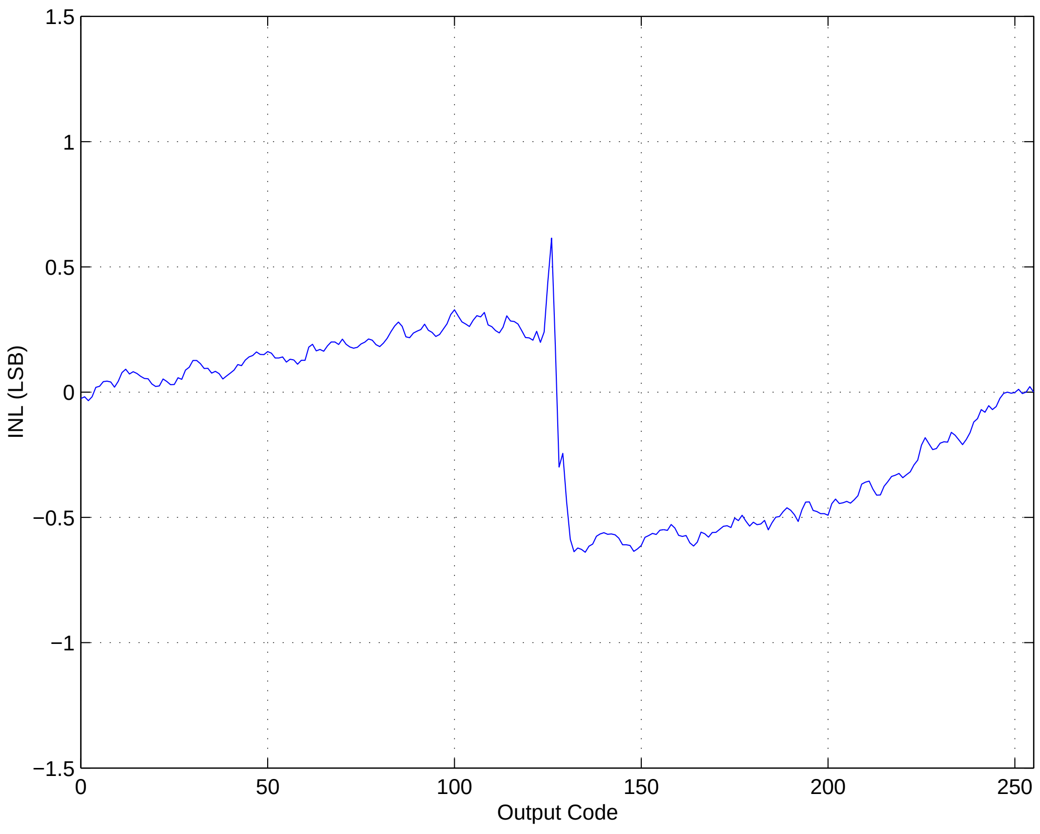


Figure 36: Integral non-linearity of the A/D.

Image Acquisition

To assess the quality of the acquired images a blank uniformly illuminated image was taken in the non-compressing mode. Figure 37 shows this raw acquired image. Ideally all the pixels in this image would have the same value. However, due to temporal noise, column and pixel FPN, the acquired image is not uniform. Although, the imager is equipped with column-level CDS circuits, some FPN is still present and that is due to a number of different reasons.

The CDS circuits remove the FPN due to the source follower column-level amplifiers but at the same time they introduce a voltage offset. Ideally, both column-level CDS circuits introduce the same offset and should cancel each other when the difference $X - \hat{X}$ is computed. However, due to variations in the parameters of the fabrication process, the offsets are not equal resulting in column FPN. Another source of column FPN is the difference between the comparators' offset voltages. On a first level, the offset of the two comparators in each A/D should cancel out each other as the same ramp is used for both voltage comparisons. This is the approach followed in [?] to reduce the FPN. At the layout level, both comparators were placed next to each other to minimize the variations of transistor threshold voltages. Despite, all these considerations, the A/D introduces an offset voltage that is different for every column. The reason maybe in part due to the offset being somewhat signal dependent and the comparator transistors not being fully matched with an inter-digitated layout. The standard deviation FPN of the image in Figure 37 is 9.055 for the column-level and 13.965 for the pixel-level case. These numbers were computed using equations (??) and (??).

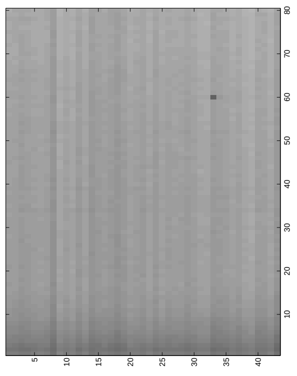


Figure 37: Raw blank test image.

The FPN observed in Figure 37 can be compensated off-line by averaging the rows a large number of blank acquired images and then subtracting this average from a new acquired image. The result of this correction is shown in Figure 38. The new FPN standard deviations are 0.163, and 0.751 for the column-level and pixel-level cases respectively.

Figures 39 and 40 show two different images acquired by the chip in the non-compressing mode.

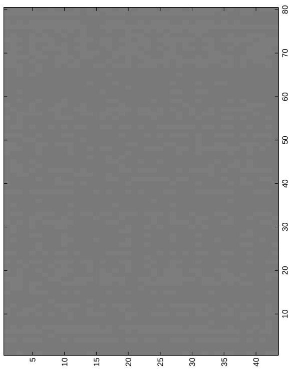
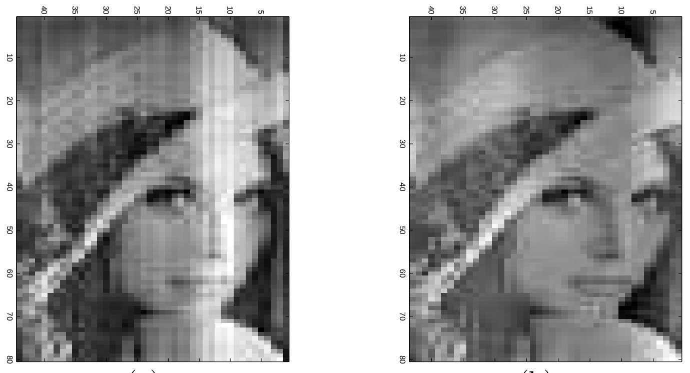
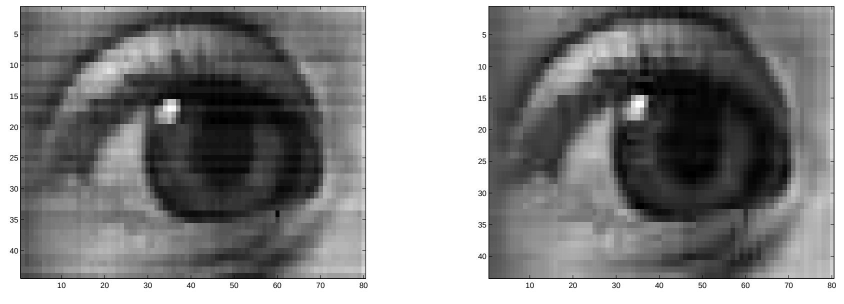


Figure 38: Compensated test image.



(a) (b)

Figure 39: Acquired *lena* image: (a) uncompensated image; (b) FPN-compensated image.

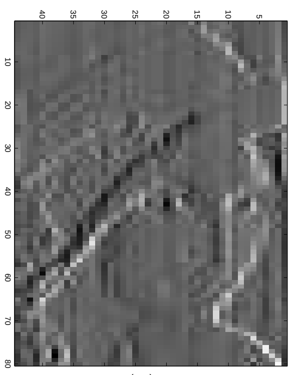


(a) (b)

Figure 40: Acquired *eye* image: (a) uncompensated image; (b) FPN-compensated image.

Image Compression

This section presents results of the chip working in still imaging compression mode. Figures 41 and 42 show the residual and the reconstructed frames of two sample images *lena* and *eye*, respectively, when the west neighbor (W) is employed as predictor.


(a)

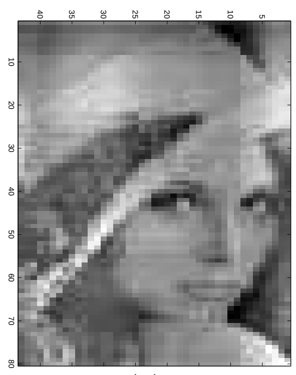
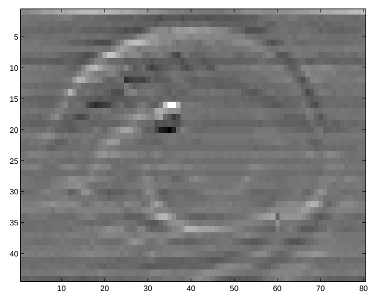

(b)

Figure 41: Sample *lena* image acquired using W neighbor as predictor: (a) residual; (b) reconstructed image.


(a)

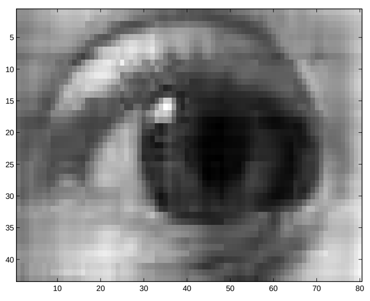
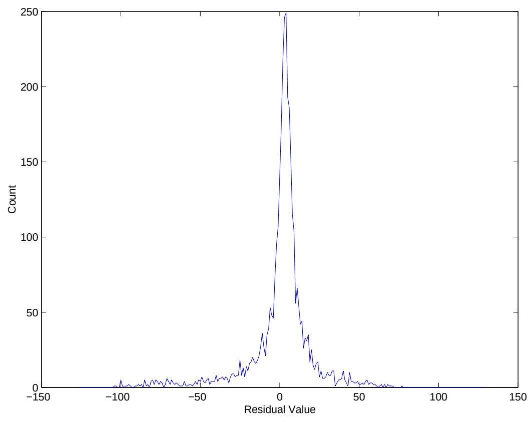
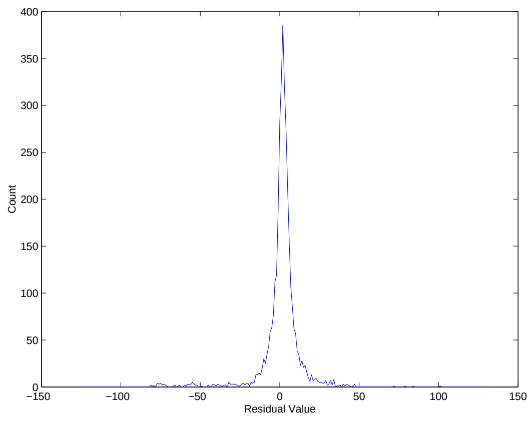

(b)

Figure 42: Sample *eye* acquired using W neighbor as predictor: (a) residual; (b) reconstructed image.

Figure 43 shows the histograms of the corresponding residual images. Based on these histograms the entropy for the *lena* image is 5.7387 bits and for the *eye* image is 4.8502. The average number of bits per pixel used by the chip to encode the *lena* residual image is 6.4099 giving a compression ratio of 1.2481 assuming each pixel is encoded using 8 bits in the uncompressed images. For the *eye* residual image the average number of bits required by the chip to encode it is 5.6250 and the compression ratio is 1.4222.



(a)



(b)

Figure 43: Histograms of the acquired residual images with W neighbor as predictor: (a) *lena*; (b) *eye*.

For comparison purposes, the reconstructed images were compressed again with the compression algorithms described in Section **??**. The results are summarized in Table 1. As can be seen from the table, the compression algorithms outperform the performance of the chip. This is due to several factors. First, the compression methods listed in Table 1 are based in the LOCO compression framework with only the entropy coder being different in each case. LOCO uses the MED predictor with context modeling and bias cancellation. Second, the mapping of equation (46) was used to map the prediction errors. Such a mapping is more efficient than the one employed in the chip which simply encodes the absolute value of the residual and appends a sign bit.

Table 1: Performance different compression algorithms on reconstructed images

File	Entropy of residual	LOCO	FELICS	Rice	Proposed
lena	5.0471	5.1824	5.1903	5.1642	5.6324
eye	3.9530	4.0315	4.0551	4.0568	4.1077

Another reason why the compression performance of computer-implemented image compression system differs even when the same adaptive algorithm is employed, is due to the manner the coding parameter is adapted. In the chip there is an independent coding parameter for each column and each one evolves independently of the others. In the computer-based implementation there is only one coding parameter for the whole image and it evolves in a raster scan fashion. Figure 44 shows the evolution of the k parameter along one of the columns of the *lena* image. It can be seen that the algorithm not only converges but also tracks the statistics of the image.

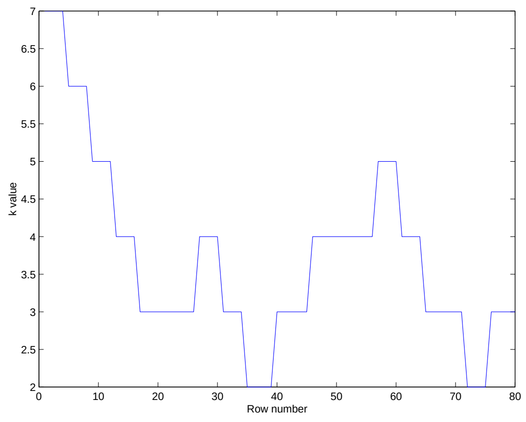


Figure 44: Adaptation of the coding parameter k.

Figures 45 and 46 show the *lena* and *eye* sample residual and reconstructed images when the north (N) pixel is used as the predictor. Figure 47 shows the corresponding histograms. The reconstructed images still show some artifacts in the form of vertical stripes. This is due to an incomplete FPN cancellation. The small FPN residuals build up due to the cumulative nature of the prediction decoding process because the column FPN is in the same direction as the prediction error when the N neighbor is employed as predictor. The compression performance of the chip when the N neighbor is used as predictor is 6.1400 bpp for the *lena* image and 5.5960 bpp for the *eye* image.

The functionality of the internal predictor selection circuit was also tested. With a threshold THR set to 7 the compression rate for the *eye* image is 1.528 and on

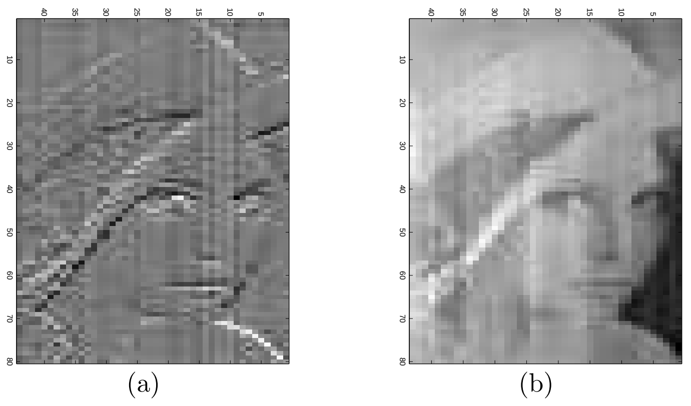


Figure 45: Sample *lena* image acquired using N neighbor as predictor: (a) residual; (b) reconstructed image.

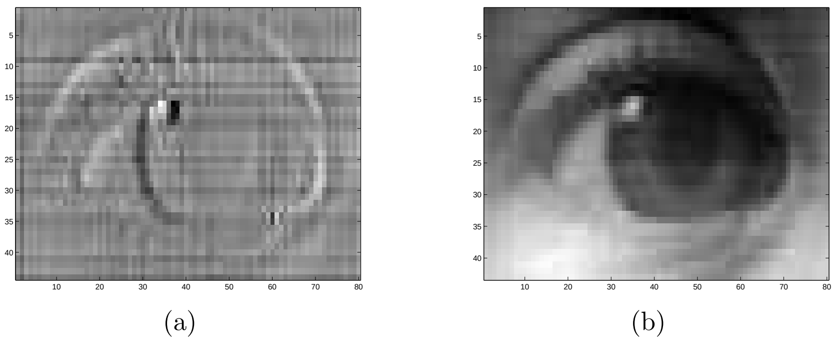
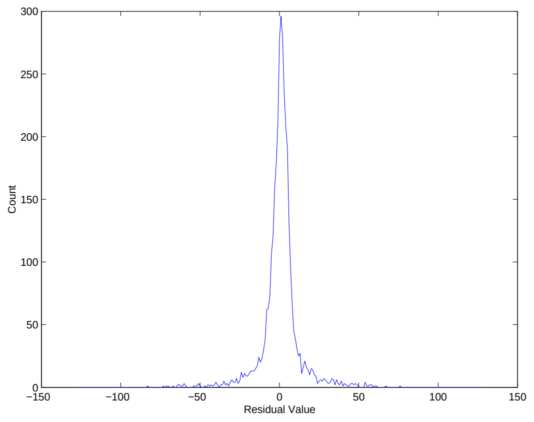
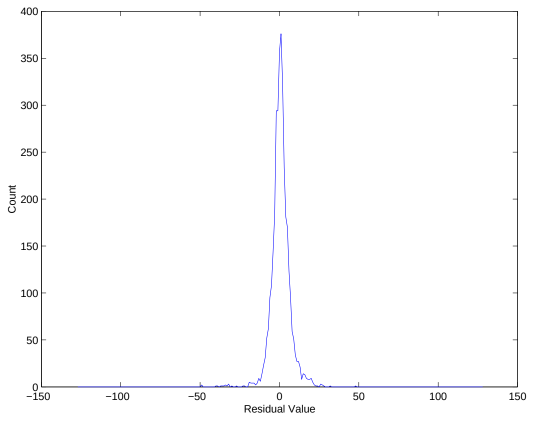


Figure 46: Sample *eye* image acquired using N neighbor as predictor: (a) residual; (b) reconstructed image.



(a)



(b)

Figure 47: Histograms of the acquired residual images with N neighbor as predictor: (a) *lena*; (b) *eye*.

average it used 5.2346 bpp. Compared with the compression performance when only one neighbor pixel was used as the predictor through out the image, an improvement of more than 0.35 bpp was obtained. Other values of the threshold THR result in a slightly inferior compression performance. Figure 48 shows the corresponding reconstructed image.

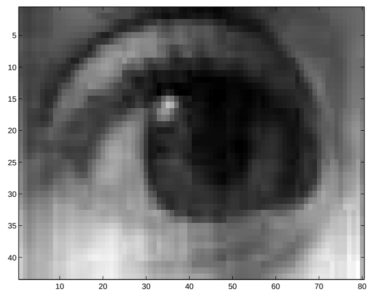
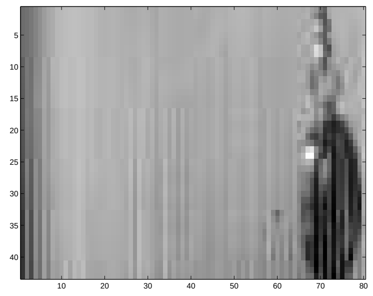
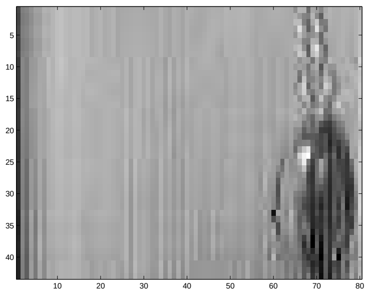


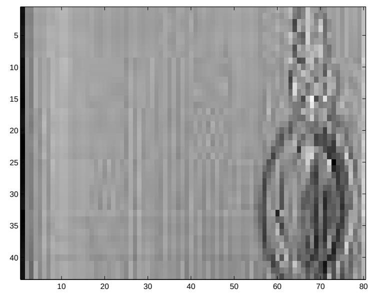
Figure 48: Reconstructed *eye* image when the predictor is adaptively selected.

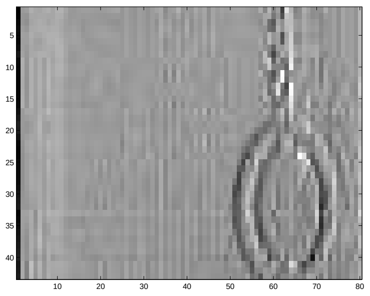
Video Compression

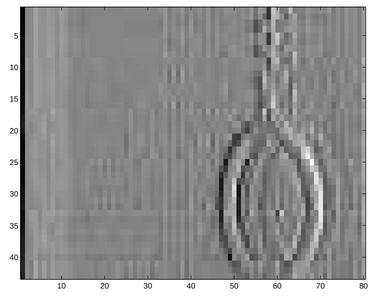
The video-mode compression was also tested and the results are shown in Figure 49. The first frame was acquired using the W neighbor as predictor. The subsequent images were acquired by externally setting the **intra/inter** line to 0. The progressive degradation and the incomplete reconstruction of the video sequence is due to charge leakage in the in-pixel capacitors. These capacitors are very small is size (150 fF) and over the integration time, which exceeds 10 ms, they discharge through the in-pixel sampling transistor. A possible solution would be to increase the in-pixel capacitor size but that will make the pixels bigger and decrease the fill factor. To keep the fill factor the same, the capacitors could be placed outside the array but that will increase the chip area. Yet another solution would be to employ low-leakage switches but that would decrease the fill factor as well. For video compression the previous frame had to be stored in some form. Because we were following an analog implementation, capacitors at the pixel level were employed. However, and due to the reasons discussed previously, the reconstructed video frames have poor quality. The compression rates obtained for the eight frames are shown in Table 2.

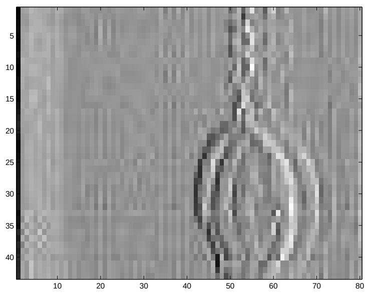

(a)

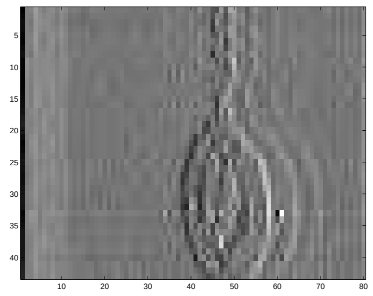

(b)


(c)


(d)


(e)


(f)


(g)

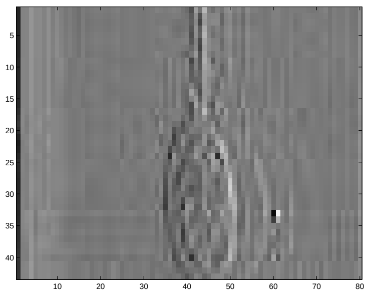

(h)

Figure 49: Acquired video frames: (a) frame 1; (b) frame 2; (c) frame 3; (d) frame 4; (e) frame 5; (f) frame 6; (g) frame 7; (h) frame 8.

Table 2: Bits/pixel obtained in the video frames

Frame	Bpp
frame 1	5.1244
frame 2	5.5977
frame 3	5.6198
frame 4	5.6948
frame 5	5.6926
frame 6	5.6375
frame 7	5.6778
frame 8	5.6173

4 Analog-to-Digital Conversion

Research efforts addressing the integration of an A/D and an entropy encoder issue have been reported in the literature. They are based on the observation that some degree of compression can be achieved in the analog-to-digital conversion process. A 12-bit A/D with a simplified Huffman encoder is presented in [35]. The design relies on a charge-redistribution digital-to-analog (D/A) converter to perform a successive approximation algorithm. To save on the number of storage elements, the input is coarsely divided into three regions, with a codeword assigned to each region. Even though it provides a sub-optimal Huffman encoding, the design reduces power consumption by not generating unnecessary bits. The complete A/D occupies an area of 3 mm^2 based on a 0.6 μm CMOS technology.

The concept of vector quantization applied to the design of an A/D is outlined in [36]. The parameters of the converters such as the hyperplanes that divide quantization cells are structured in the form of a binary tree. The design includes a neural network-based learning rule that adaptively updates the boundaries of the quantization cells. The complexity of the design grows with the dimensionality of the input vector as it requires one scalar A/D per dimension and the respective storage space for the binary tree.

In this research work a novel integration of two A/D structures with a Golomb-Rice encoder is proposed. The integrations are possible by noting the common circuitry between A/Ds and the Golomb-Rice encoder. In the first A/D architecture, an integrating A/D, a digital counter is employed to quantize the input and to simultaneously generate the corresponding Golomb-Rice codeword. This integration results in a very linear A/D that is also able to perform non-uniform quantization. The validity of this design has been verified through transistor level simulations. The second architecture examined is a cyclic or algorithmic A/D. The inner loop of this type of converter can be modified to generate the unary part of the Golomb-Rice codeword. This converter also provides shorter conversion times.

Joint Quantization/Coding of a Laplacian Source

The proposed A/D jointly performs the quantization and encoding of its input. Consider a zero mean random variable x with a Laplacian probability distribution function (pdf) $f_X(x)$, i.e.:

$$f_X(x) = \frac{1}{\sqrt{2\sigma^2}} e^{-\frac{\sqrt{2}}{\sigma}|x|} \tag{30}$$

where σ^2 is the variance of x. This random variable will be uniformly quantized based on the scheme shown in Fig. 50, where Δ is the quantization step size. The

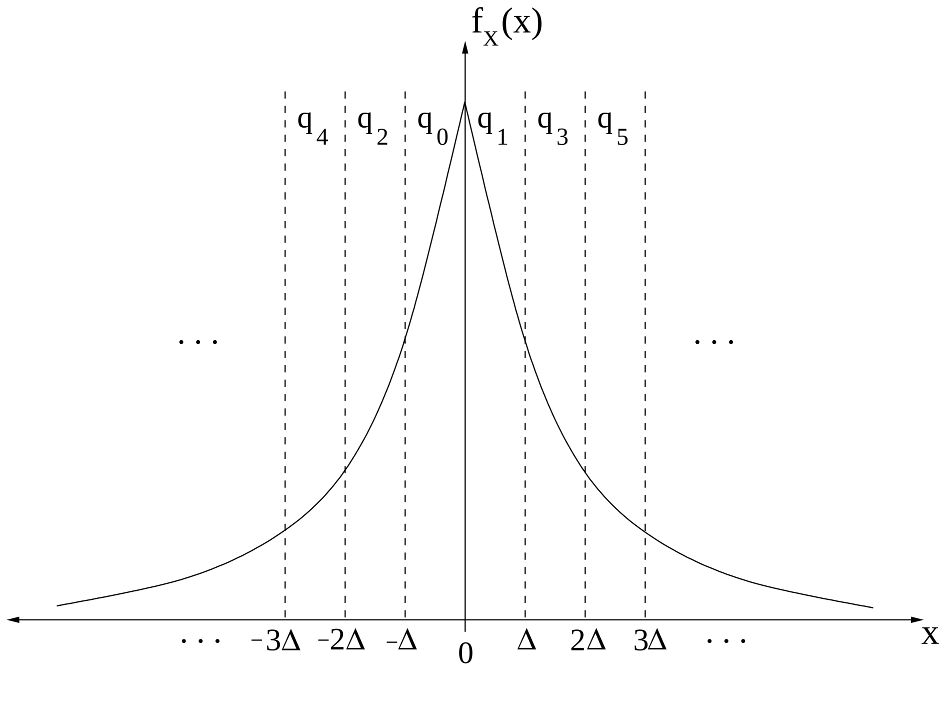


Figure 50: Uniform quantization of a Laplacian distributed signal.

probability of x being in a quantization interval q_{2i+1}, ($i = 0, 1, 2, \cdots$), is readily obtained by integrating the pdf:

$$p(q_{2i+1}) = \int_{i\Delta}^{(i+1)\Delta} \frac{1}{\sqrt{2\sigma^2}} e^{-\frac{\sqrt{2}}{\sigma}x}\, dx\ , \quad i = 0, 1, 2\cdots \tag{31}$$

Evaluating this integral

$$p(q_{2i+1}) = \tfrac{1}{2} e^{-\frac{\sqrt{2}}{\sigma}i\Delta}(1 - e^{-\frac{\sqrt{2}}{\sigma}\Delta})\ , \quad i = 0, 1, 2\cdots \tag{32}$$

Notice that $p(q_{2i}) = p(q_{2i+1})$. Letting $p_o = e^{-\frac{\sqrt{2}}{\sigma}\Delta}$, one can write $p(q_{2i}) = p(q_{2i+1}) = \frac{1}{2}(1-p_o){p_o}^i$. If the quantization intervals q_{2i} and q_{2i+1} are jointly represented by the codeword $\tilde{c}_i$, the probability of occurrence of this codeword is $(1-p_o){p_o}^i$. Since this is now a geometric distribution, the optimal assignment for $\tilde{c}_i$ is the one provided by the Golomb encoding algorithm [44]. A sign bit b_s is necessary to uniquely determine the quantization interval in which the input x lies. The sign bit will be set to "1" if x is in the interval q_{2i+1} and set to "0" if x is in the interval q_{2i}. The final codeword is simply the concatenation of $\tilde{c}_i$ and b_s. This assignment essentially uses the LSB of the codeword as a sign bit. The symmetry of the distribution around zero in Fig. 50 ensures this coding is still optimal [59].

To obtain a simple estimate of the optimal parameter m needed in the algorithm, recall the Golomb condition ${p_o}^m = \frac{1}{2}$. Taking the logarithm of both sides and restricting to the special case $m = 2^k$ yields

$$k = \left\lceil \log_2 \left(\frac{\log 2}{\Delta} \frac{\sigma}{\sqrt{2}} \right) \right\rceil \tag{33}$$

where $\lceil x \rceil$ is the smallest integer greater than or equal to x. Notice also that the expected value of $|x|$ is given by

$$E\{|x|\} = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\sigma^2}} |x| e^{-\frac{\sqrt{2}}{\sigma}|x|}\, dx = \frac{\sigma}{\sqrt{2}} \tag{34}$$

Combining this result with (33) gives

$$k = \lceil \log_2 (B \cdot E\{|x|\}) \rceil \tag{35}$$

with $B = \frac{\log 2}{\Delta}$. Equation (35) shows a simple way to estimate the value of k. The parameter k can be chosen to match the variance, σ^2, of the input. For some applications this is known *a priori*, for others this can be estimated and programmed.

The hardware implementation of equation (35) is not trivial as we will have to compute the average of the analog input, multiply the result by a constant, take the logarithm and convert the result into a digital representation. A first step toward the simplification of the computation of k is to realize that the average, $E\{|x|\}$, can be approximated by computing the average of the output of the A/D. This has the added benefit that all subsequent operations will be carried out in the digital domain. To simplify the hardware even further, only the last 64 samples are employed in the computation of the average and the constant $B = \frac{\log 2}{\Delta}$ will be ignored. Finally, the logarithm function can be implemented with a look-up table. A similar strategy was used in [10]. Although this reduces the hardware complexity, in some applications (e.g., on-sensor processing) less complex schemes will be needed. In Section 5 a low-complexity algorithm for the adaptive computation of k will be presented.

A/D Architecture

The Golomb-Rice encoding can be performed by a binary counter. The encoding steps are summarized as follows:

```
1) Reset the counter ;
2) If counter value equals input $n$ then go to 6) ;
3) Increment counter by one;
4) If the $k^{th}$ bit of the counter, $b_{k-1}$, goes from
   "1" to "0", output a "1" ;
5) Go to 2) ;
6) Output a "0" followed by counter bits $b_0 \cdots b_{k-1}$;
```

Step **4)** generates the unary code for $\lfloor n/2^k \rfloor$. The counter used to generate the Golomb-Rice codes can actually be shared with an integrating A/D. This key concept is conceptually illustrated in Fig. 51, where the shift register is used to store the unary code. Figure 52 shows the proposed architecture in more detail.

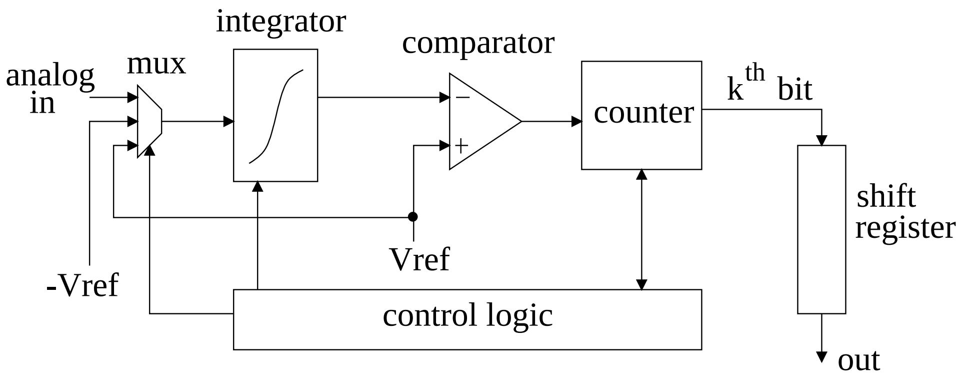


Figure 51: Conceptual block diagram of proposed ADC.

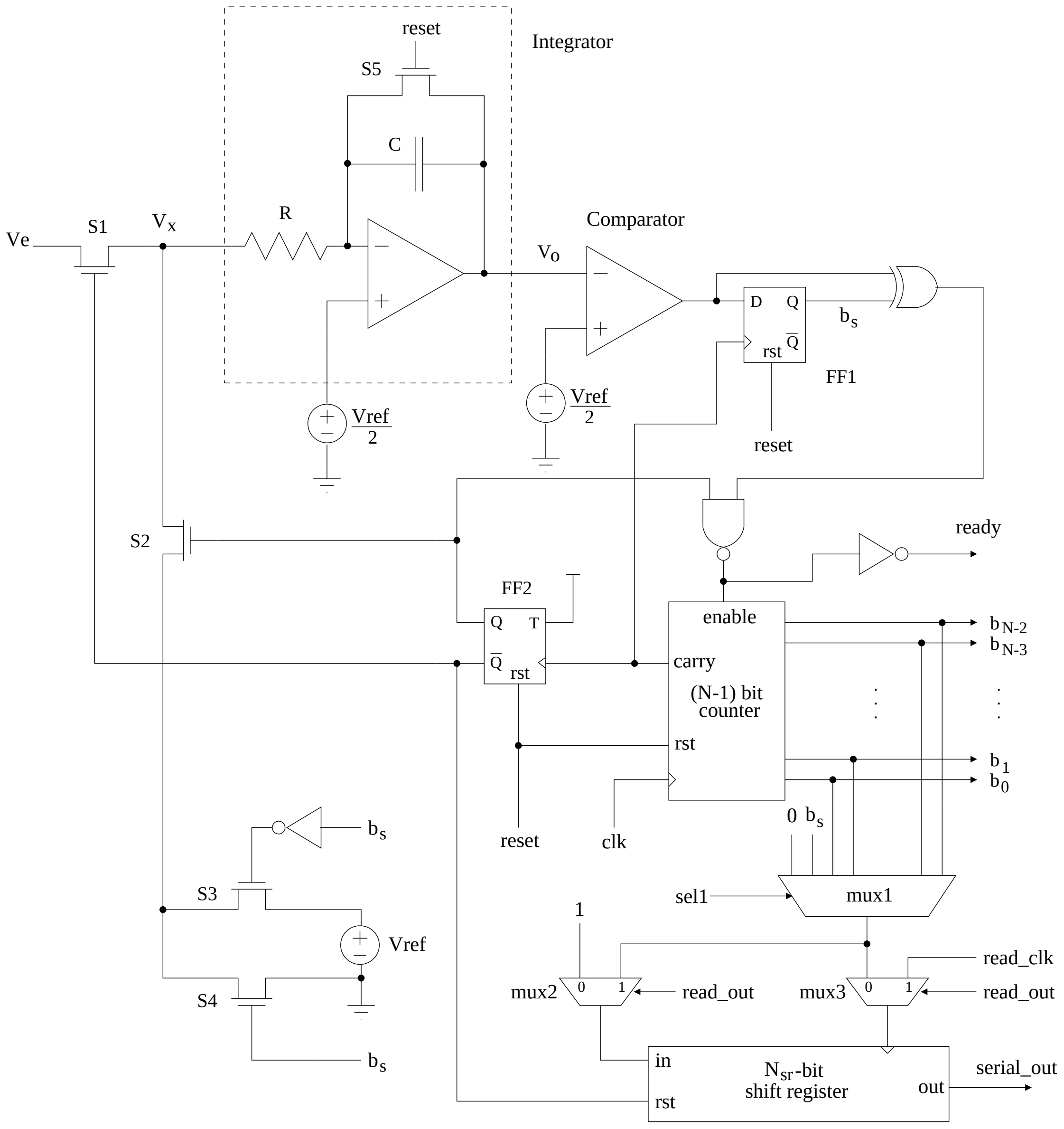


Figure 52: Block diagram of the A/D.

The main building blocks of the dual-slope integrating A/D with compressed output are: a voltage integrator, a comparator, an $(N-1)$-bit digital counter, a negative-edge N_{sr}-bit shift register, and a $(N+1)$-to-1 multiplexer. It is assumed in this design that the input signal V_e lies in the range from 0 to V_{ref} volts and it is centered around $\frac{V_{ref}}{2}$. Hence, it can be expressed as $V_e = \frac{V_{ref}}{2} + v_e$, where $-\frac{V_{ref}}{2} \leq v_e \leq \frac{V_{ref}}{2}$. The A/D has a resolution of N bits with a quantization step size Δ given by $\frac{V_{ref}}{2^N}$ when uniform quantization is performed. The final codeword is read out serially at the output of the shift register.

The conversion cycle starts when the **reset** signal goes low. The conversion has two phases. In phase I the switch **S1** is closed and the switch **S2** is open allowing the integrator to integrate V_e. V_o, the output of the integrator during this phase, is given by

$$V_o(t_1) = \frac{V_{ref}}{2} - \frac{1}{RC}\int_0^{t_1} v_e\, dt = \frac{V_{ref}}{2} - \frac{v_e t_1}{RC} \tag{36}$$

where t_1 is the time since the beginning of the conversion. Simultaneously, the digital counter counts up. After 2^{N-1} clock cycles, the carry output of the counter is pulsed, toggling the T flip-flop FF2, which in turn opens the switch **S1** and closes the switch **S2**. This event marks the end of phase I and the beginning of phase II. At this point the voltage at the output of the integrator is given by

$$V_{o,I} = \frac{V_{ref}}{2} - \frac{2^{N-1} T_{clk}\ v_e}{RC} \tag{37}$$

where T_{clk} is the period of the counter clock.

The rising edge of the carry signal also stores the output of the comparator on the D flip-flop FF1. The Q output of FF1 is the sign bit and will constitute the least significant bit of the conversion b_s. If $v_e > \frac{V_{ref}}{2}$, b_s is set to "1", otherwise it is set to "0". Notice that this bit reflects the polarity of v_e and also assigns even values to inputs with negative v_e and odd values to inputs with positive v_e (this is consistent with the notation used in Fig. 50). The voltage V_x is set to ground or V_{ref} through the switches **SW3** and **SW4** depending whether b_s is set to "1" or "0". The output of the integrator in phase II is given by

$$V_o(t) = \begin{cases} V_{o,I} + \frac{1}{RC}\frac{V_{ref}}{2} t_2 & \text{if } v_e > 0; \\ V_{o,I} - \frac{1}{RC}\frac{V_{ref}}{2} t_2 & \text{if } v_e \leq 0. \end{cases} \tag{38}$$

where t_2 is the time since the beginning of phase II. After the generation of the carry signal, the counter value is set to zero and the counting process starts over again. To generate the unary code, the selection input **sel1** of the multiplexer **mux1** is set to $k+1$, with k being the Golomb-Rice code parameter described previously. During conversion, the **read_out** input is set to "0". This setting causes the shift register to shift in a "1" every time counter bit b_{k-1} goes from "1" to "0".

The conversion cycle ends when V_o reaches $\frac{V_{ref}}{2}$ causing the comparator output to flip. The logic formed by the XOR and NAND gates detects this change and stops the counter. Fig. 53 shows graphically the waveform of V_o for $v_e > 0$ V during the two phases of the conversion process. In the referred figure, $T_1 = 2^{N-1}T_{clk}$. Solving

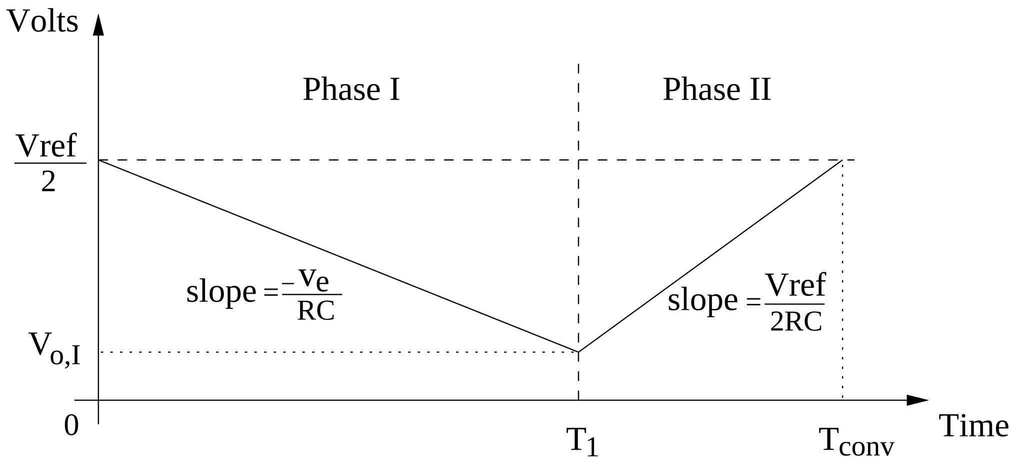


Figure 53: Integrator output waveform.

(38) for t_2 with the condition at the end of conversion, $V_o(t) = \frac{V_{ref}}{2}$, yields

$$T_{conv} = \frac{|v_e|}{\frac{V_{ref}}{2}} 2^{N-1} T_{clk} + T_1 \tag{39}$$

The value of the counter at the end of the conversion is also a measure of the time $T_{conv} - T_1$ or equivalently,

$$2^{N-2}\, b_{N-2} + \cdots + 2^1\, b_1 + 2^0\, b_0 = \left\lfloor \frac{T_{conv} - T_1}{T_{clk}} \right\rfloor \tag{40}$$

Thus, the binary word $b_{N-2}b_{N-3}\cdots b_1 b_0$ is the digital representation of $|v_e|/(V_{ref}/2)$. The unary code for $\lfloor n/2^k \rfloor$, with $n = 2^{N-2}b_{N-2} + \cdots + b_0$, is already in the shift register and the binary code for (n mod 2^k) is simply $b_{k-1}\cdots b_0$. The output signal **ready** is set to "1" signaling the end of conversion.

The next step is the read out process. In this step, the input **read_out** is set to "1" causing the output of the multiplexer **mux1** to be redirected to the shift register input through **mux2**. By successively incrementing the selection input of **mux1** and toggling the input **read_clk** the remaining part of the Golomb-Rice code is shifted into the register. The codeword is available in a serial fashion at the output of the shift register **serial_out**. The converter output has the form "$11 \cdots 10$", $b_s b_0 b_1 \cdots b_{k-1}$. The choice of the length N_{sr} of the shift register is a trade-off between hardware complexity, input range, and compression ratio. If the shift register is too short for some values of the input v_e, the register will get full before the end of the conversion, limiting the input range in this manner. The range could be increased by increasing the value of the parameter k, reducing the length of the unary code, but an increment of k beyond its optimal value will impact the compression ratio negatively.

Hardware Implementation

The digital counter considered in this design is a ripple counter due to its complexity advantages. The comparator is a latched comparator with dynamic latches at its outputs to hold the previous comparison result. A continuous-time integrator has been utilized in the realization of dual-slope converter due to its simplicity. The digital section has a gate count of less than 100. The overall converter architecture including the passive components occupies a layout area of 0.09 mm^2 based on a 0.5 μm CMOS target technology. The presented design has been verified at the circuit level using the target technology. In particular, the circuit implementation of Fig. 52 was simulated at the transistor level in Hspice for a typical speech or audio application with a conversion resolution of $N = 8$ bits. The counter clock frequency has been set at 20 MHz which in the worst case gave a conversion speed of 73,000 conversions/sec including the read out overhead.

Fig. 54 shows the serial output of the converter. In this simulation, the parameter k was set to 3 and $V_e = 2.88V$. Considering that in this case $v_e = 2.88 - 2.5 = 0.38$ V, equations (39) and (40) yield a counter value of '0010011' in binary (19 in decimal) which is further verified by the simulation plot in Fig. 54. Due to complexity advantages in the layout, the inverted version of the counter output bits are read out. To demonstrate the validity of the conversion, all the counter bits are shown in the figure although it is only required to read out $k = 3$ counter bits. The output codeword in this case would be '110', $b_s b_0 b_1 b_2$.

Assuming correct digital function, the output of the proposed architecture would be identical to that of a standard dual-slope A/D followed by a general purpose digital circuit for entropy coding. No additional errors are incurred with the combination of analog and digital functions. The accuracy of the proposed converter is, thus, the same as that of a dual-slope converter. Since the proposed converter is based on a dual-slope architecture and it is targeting an accuracy of 8 bits, error correction techniques typically employed in high-resolution cyclic and pipeline converters were

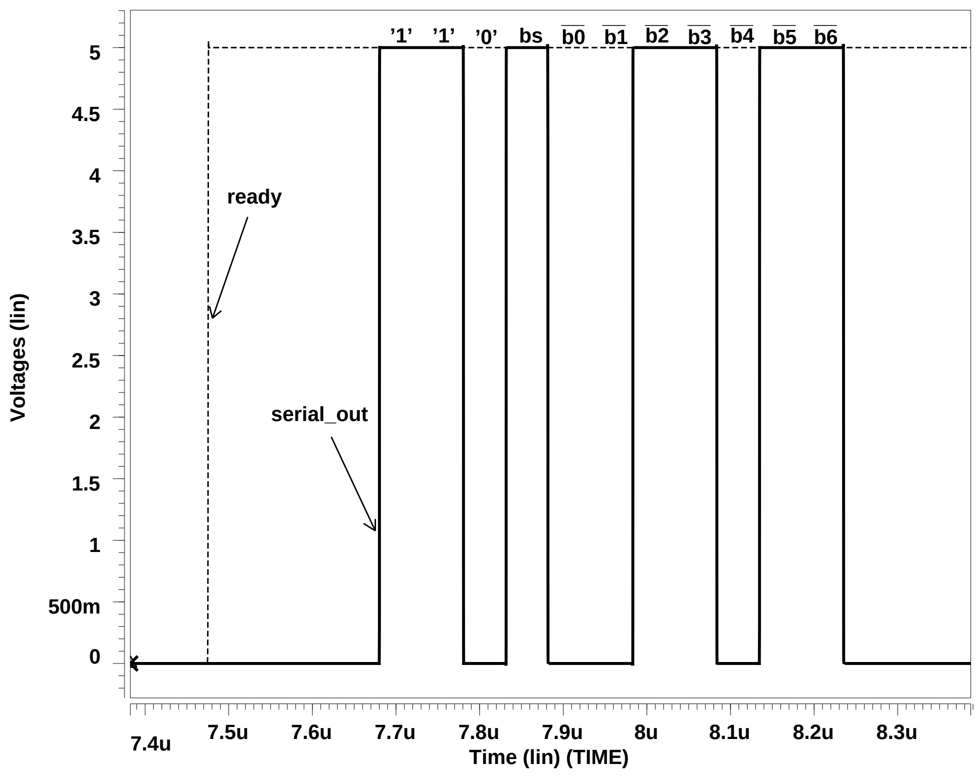


Figure 54: Simulation waveforms showing the output of the converter.

deemed unnecessary. Moreover, the effects of circuit non-idealities such as component matching are greatly reduced due to the dual slope operation of the converter.

To appreciate the complexity savings of the presented design, consider a standard implementation consisting of a regular dual-slope A/D followed by an independent Golomb-Rice encoder. To generate the unary code in one clock cycle after the A/D conversion, a barrel shifter and some control logic are needed. A data formatter circuit is also required to concatenate the k LSB bits of the converter to the unary code. A circuit like this has already been implemented in a 1 μm CMOS technology and is reported in [54]. It occupies a silicon area of 2.91 mm^2. In contrast, in the proposed approach a multiplexer, an 8-bit shift register, and some control logic are needed besides the dual-slope converter. The area required to implement this extra circuitry in a 0.5 μm CMOS technology is 7819 μm^2 which, after scaling, is equivalent to 0.0313 mm^2 in a 1 μm CMOS technology. Consequently, integration of the A/D and the Golomb-Rice coder results in significant complexity reduction.

Regarding power consumption of the proposed converter, the multiplexer consumes a small amount of power because the value of k is fixed during a conversion cycle. The contribution of the counter is already part of the power consumption of the dual-slope converter. Thus, the most significant portion of extra power consumption arises from the operation of the 8-bit shift register which is clocked at a frequency of $\frac{1}{2^k\ T_{clk}}$.

Single-Slope Analog-to-Digital Conversion

A drawback of the dual-slope converter is its long conversion time. Fortunately, the conversion time can be reduced almost by half while keeping all other advantages. This is accomplished with a single-slope A/D. In a single-slope A/D the input voltage is compared with a voltage ramp. The time it takes for the ramp to reach the input is quantized by a counter. The unary code is generated in the same way - by employing a shift register. This type of converter and its implementation is described in detail in Section 2. A single-slope converter still requires a time $O(2^N)$ clock cycles to perform a conversion. Another converter, the cyclic A/D, can significantly reduce the conversion time and still generate Golomb-Rice codes at its output.

Cyclic Analog-to-Digital Conversion

The architecture of a cyclic or algorithmic A/D is presented in Figure 55. It consists of a comparator, an amplifier with gain 2, and two sample-and-hold circuits. The operation of a cyclic A/D is based on the circulation of a residual voltage through the loop formed by the sample-and-hold circuits and the amplifier [39]. Each pass through the loop generates one bit. The MSB is output first. In principle the residual can be circulated as many times as bits are needed. In practice, however, the mismatches and noise of the analog circuits limit the resolution the this converter to 8 to 10 bits. Digital correction techniques can be employed to obtain up to 15 or 16 bits of resolution [40]-[42].

In the first pass the analog input, v_{in}, is sampled and compared with the reference voltage $V_{ref}/2$. At this point $V_x = v_{in}$. The output of the comparator determines the output bit. In each pass around the loop the value of the output bit is given by

$$b_i = \begin{cases} 1 & \text{if } V_x \geq \frac{V_{ref}}{2}; \\ 0 & \text{if } V_x < \frac{V_{ref}}{2} \end{cases} \tag{41}$$

If $b_i = 1$ the switch SW2 is switched up, otherwise is switched down. After the first pass the switch SW1 is switched to its down position. On the second pass the voltage

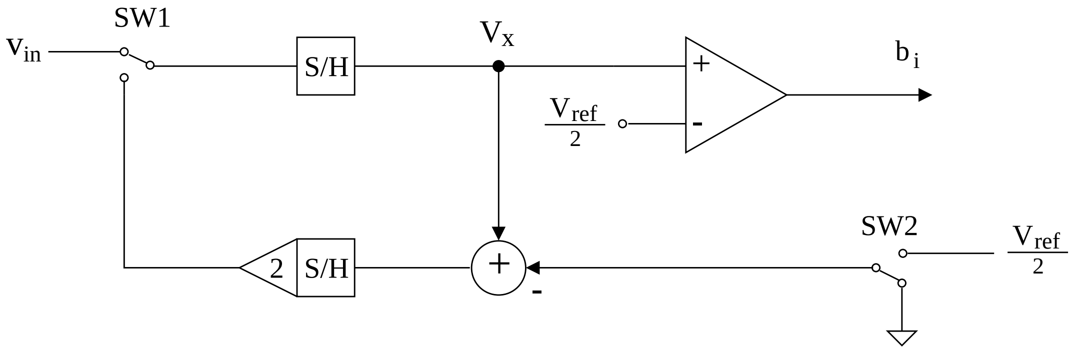


Figure 55: Block diagram of a cyclic A/D

V_x is given by

$$V_x(2) = 2(V_x(1) - b_1\frac{V_{ref}}{2}) = 2V_x(1) - b_1V_{ref} \tag{42}$$

and on the third pass is given by

$$V_x(3) = 2(V_x(2) - b_2\frac{V_{ref}}{2}) = 4V_x(1) - 2b_1V_{ref} - b_2V_{ref} \tag{43}$$

In general, for $n+1$ pass (recalling that $V_x(1) = v_{in}$)

$$V_x(n+1) = 2^n v_{in} - V_{ref}(2^{n-1}b_1 + 2^{n-2}b_2 + \cdots + 2^0 b_n) \tag{44}$$

Thus, the bits $b_1, b_2, \cdots, b_n$ are a digital representation of the input v_{in}

The structure of the cyclic A/D can be slightly modified to perform the analog-to-digital conversion and generate the corresponding Golomb-Rice code simultaneously. To generate the unary code, the gain of the amplifier needs to be changed to unity and the reference voltage $V_{ref}/2$ to $V_k = V_{LSB}2^k$ as shown if Figure 56. As before, the input v_{in} is sampled in the first pass and then the switch SW1 is switched down. In every subsequent pass through the loop the voltage V_k is subtracted from the residual voltage V_x. This configuration is maintained as long as $V_x > V_k$ or equivalently $u_i = 1$. Whenever $V_x \leq V_k$, output a **0** and return to the configuration of Figure 55. With the switch SW1 in the down position iterate the circuit k times to obtain the remaining bits of the codeword.

The advantage of employing a cyclic A/D is that the operations are carried out in the analog domain, thus, more compact circuits can be employed. Another advantage is that the conversion time is reduced compared to the integrating A/D. However, the exact conversion time depends on the input and the value of k. If the input is large and k is small, a long unary code will be generated which means a long conversion time. On the other hand, if the input is small and k is large only few bits are generated

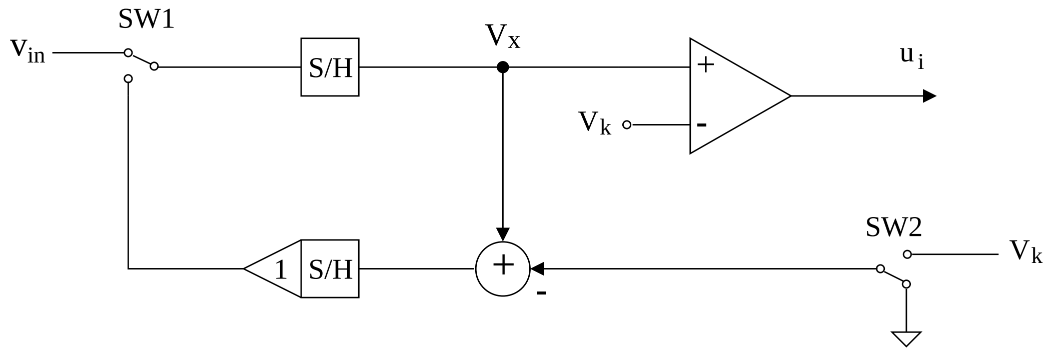


Figure 56: Block diagram of a modified cyclic A/D

and the conversion time is very short. On average, though, the conversion time will be less than the typical N cycles required by a standard cyclic A/D.

A drawback of this type of architecture is that in each pass around the analog loop, noise and the effects of component mismatches are accumulated precluding the accurate generation of long unary codes. Another drawback is that an accurate generation of the voltages V_k is not trivial. One solution to these drawbacks could be to limit the length of the unary codes and to use offset compensation techniques in the comparator.

5 Low-Complexity Entropy Encoding

Entropy coding algorithms are essential building blocks in data compression. Entropy coders remove the redundancy in a sequence of symbols by encoding frequent symbols with fewer bits than those used to encode less frequent symbols.

In this section, a low-complexity adaptive algorithm for Golomb-Rice codes is proposed. Due to its complexity advantages, the proposed algorithm is suitable for on-sensor integration. The algorithm does not require complex arithmetic operations, accumulators or data buffers and is implemented around a simple up/down counter. The algorithm works in a greedy fashion, incrementing or decrementing the value of the Golomb-Rice code parameter according to the statistics of the input sequence. The performance of the algorithm is analytically determined by modeling the behavior of the up/down counter with a Markov chain. The compression performance is comparable to other state-of-the-art adaptive Golomb coders used in image, audio and scientific data compression.

The Golomb-Rice Codes

The Golomb-Rice codes form a family of codes parameterized by an integer $m > 0$. The encoding of a non-negative integer n proceeds as follows: represent n as $n = qm + r$ where $q = \lfloor n/m \rfloor$ and $\lfloor y \rfloor$ is the greatest integer less than or equal to y. The quotient q is encoded with a unary code and the remainder r is encoded with a Huffman code with codewords of lengths $\lfloor \log_2 m \rfloor$ if $r < 2^{\lfloor \log_2 m+1 \rfloor} - m$ or lengths $\lfloor \log_2 m \rfloor + 1$ otherwise. When the source of integers follow a geometric distribution, i.e. $P(n) = (1-\theta)\theta^n$, the optimum choice of m satisfies the inequality: $\theta^m + \theta^{m+1} \leq 1 < \theta^m + \theta^{m-1}$ [44].

When m is a power of two, i.e. $m = 2^k$, $k = 0, 1, 2, \cdots$, the encoding is a very simple procedure. As before, the quotient q is encoded with a unary code but the reminder r is encoded with a natural binary code. When the input n is in binary format, the codeword consists simply of the k least significant bits of n preceded by the unary code of the higher order bits of n [43], [55], [10]. The length of the codeword is

$$l_{n,k} = k + 1 + \lfloor n/m \rfloor \tag{45}$$

To encode positive and negative integers x, a suitable mapping is defined as follows [10]:

$$M(x) = \begin{cases} 2x & \text{if } x \geq 0 \\ 2|x| - 1 & \text{if } x < 0 \end{cases} \tag{46}$$

An alternative approach is to encode the absolute value of the input with a Golomb code and append a sign bit for nonzero values [57]. It has been shown that, under certain conditions, the Golomb-Rice codes are optimal for sources with a Laplacian distribution [56].

Adaptive Algorithms

The performance of the Golomb-Rice codes depends on how the parameter k is chosen. Basically, the problem of choosing the coding parameter k can be stated as follows: based on the past sequence of inputs $x^t = x_1 x_2 \cdots x_t$, select a k such that it minimizes the output length. Mainly, there are two approaches to this problem. The first approach tries to estimate the distribution parameters from x^t and minimize the average code length [45]. The second approach is an exhaustive search of the coding parameter k that performs best on x^t [49], [55]. Following the first approach the Maximum-Likelihood (ML) estimation approach has been used in [46] resulting in the following theorem.

Theorem 1: Let $\phi = (\sqrt{5}+1)/2$ ("golden ration"). Encode x_{t+1} according to the following rules.

1) If $S_t \leq \phi t$, compare S_t, $t - N_t$, and N_t. If S_t is largest, choose code Γ_1. Otherwise, if $t - N_t$ is largest, choose Γ_0. Otherwise, choose Γ_0'
2) If $S_t > \phi t$, choose code Γ_{k+1}, $k \geq 1$, provided that

$$\frac{1}{\phi^{2^{-k+1}} - 1} < \frac{S_t}{t} \leq \frac{1}{\phi^{2^{-k}} - 1} \tag{47}$$

where

$$S_t = \sum_{i=1}^{t} (|x_i| - g(x_i)), \qquad N_t = \sum_{i=1}^{t} g(x_i) \tag{48}$$

where $g(x) = 1$ if $x < 0$, or 0 otherwise, $\Gamma_k(x)$ is the Golomb-Rice code of $M(x)$, Γ_0 is the Golomb-Rice code of $M(x)$ with $k = 0$, and Γ_0' is the Golomb-Rice code of $M'(x)$ with $k = 0$. The function $M(x)$ is the integer mapping defined in equation (46) and $M'(x) = M(-x-1)$.

Based on the observation that the inverse of the natural logarithm of the golden ratio is very close to 2 ($1/\ln\phi \approx 2.078$), the authors of [46] propose a low-complexity approximation of Theorem 1 as follows.

Let $S_t' = S_t + (t/2) - (t/8)$.
1) If $S_t' \leq 2t$, compare S_t, $t - N_t$, and N_t. If S_t is largest, choose code Γ_1. Otherwise, if $t - N_t$ is largest, choose Γ_0. Otherwise, choose Γ_0'

2) If $S'_t > 2t$, choose code Γ_{k+1}, $k \geq 1$, provided that
   $t2^k \leq S'_t < t2^{k+1}$

In JPEG-LS this adaptation rule is further simplified by noting that $S_t + N_t$ is the accumulated sum of the magnitudes of the inputs x. This accumulated sum is stored in a register A. Another register, B, is used to store the sum of the inputs x. Finally, a counter N keeps the number of input symbols x processed so far. These simplifications yield the following algorithm [45]

a) Compute k as
   $k = \min\{k' | 2^{k'} N \geq A\}$.
b) If $k > 0$ code Γ_k. Otherwise, if $k = 0$ and $2B > -N$ choose code Γ_0. Otherwise, choose code Γ'_0.

This value of k can be computed in C language with the following one-line statement [10]:

```
for( k=0; (N<<k)<A; k++ );
```

To improve the adaptation to the changing statistics of the source, the values of the registers A and B are halved every time N reaches a predetermined threshold N_o. Typical values of N_o are between 64 and 256. In an earlier version of LOCO [10] k was estimated with the following equation

$$k = \lceil \log_2 E\{|x|\} \rceil \tag{49}$$

where $E\{|x|\}$ denotes the expected value of the magnitudes of the input. Similar estimates have been used in [53] and [50], i.e.

$$k = \lfloor \log_2 E\{|x|\} + C_1 \rfloor \tag{50}$$

where $C_1 \approx 0.97$ is a constant. It turns out that the value of k in equation (49) can also be computed with the C language "one-liner" presented above. Behind the simplicity of the "one-liner" is the assumption that a microprocessor will be available in the target application. In the cases where this might not be true, the LOCO adaptation rule will have to be implemented by a dedicated hardware. The dedicated hardware will have to include the register A, a counter for N, and a control circuit implementing the `for` loop. The control circuit will need to be able to perform shift operations, comparisons and increments of 1.

In [55] a full search of the coding parameter k is carried out. The input sequence is divided into blocks of length J symbols. Each block of data is encoded in parallel by a set of variable-length codes. Each variable-length code is referred to as a coding option and corresponds to a specific value of the parameter k. The coding option for $k = 0$ is called the Fundamental Sequence or FS. The other options are called split-sample options and correspond to values of k greater than zero. The coding option that provides the highest compression is selected to encode the block of data. An ID bit pattern that identifies the winning option is added at the beginning of the compressed sequence for each block. This procedure is repeated for each block of length J effectively achieving adaptation to the changing statistics of the source. A typical value of the parameter J that results in good compression performance is 16. In the CCSDS Recommendation the first two options are reserved for low entropy cases and the last option is a non-compression option. The length of the ID identifier can be set to 3 or 4 bits depending on whether the dynamic range of the input symbols is 8 or 16 bits [58]. The Rice algorithm has been implemented on a VLSI chip in [54]. The encoder chip contains 37,000 transistors and occupies an area of 5x5 mm^2 in a 1.0 μm CMOS technology. The encoder chip includes a block that generates the Golomb-Rice codes, a DPCM+Mapper, and a Data Formatter block. The selection of the code parameter for each block of J samples is performed by a circuit that counts the number of bits required for each coding option, a circuit that selects the winning option and a FIFO buffer of 56 bytes to hold the input data while the option selection is performed. These three circuits combined occupy approximately 39% of the encoder's hardware.

Another full search approach is presented in [49] as part of an image compression system dubbed FELICS. FELICS keeps a cumulative total, for each value of k, of the codeword length that would have resulted if the parameter k was used in all the input symbols processed so far. The coding parameter with the smallest cumulative length is used to encode the next input symbol. The decoder can perform the same decision based on the symbols decoded so far. The algorithm quickly converges to a good estimate. On average, this algorithm will generate at most $O(\sqrt{N})$ more bits than the average number of bits generated if the best parameter was used [49]. The hardware implementation of this algorithm requires an accumulator for each possible value of k and a control logic that selects the accumulator with the minimum value. The FELICS algorithm is an example of a "plug-in" algorithm where the distribution parameters of the input source are not estimated, instead the coding parameter that has been the best so far is employed.

Proposed Adaptive Algorithm

A "plug-in" low-complexity algorithm that is able to adaptively select the Golomb-

Rice code parameter k is described in this Section. The algorithm operates in a feedback loop as shown in Figure 57. The algorithm monitors the length of the output codewords and in turn increments or decrements the value of the code parameter k to minimize the length of the compressed sequence. The decoder can follow the parameter changes based on the last decoded codeword. Thus, there is no need to transmit any side information. The input M is a threshold and its function is explained below. Its value is known *a priori* to both the encoder and to the decoder.

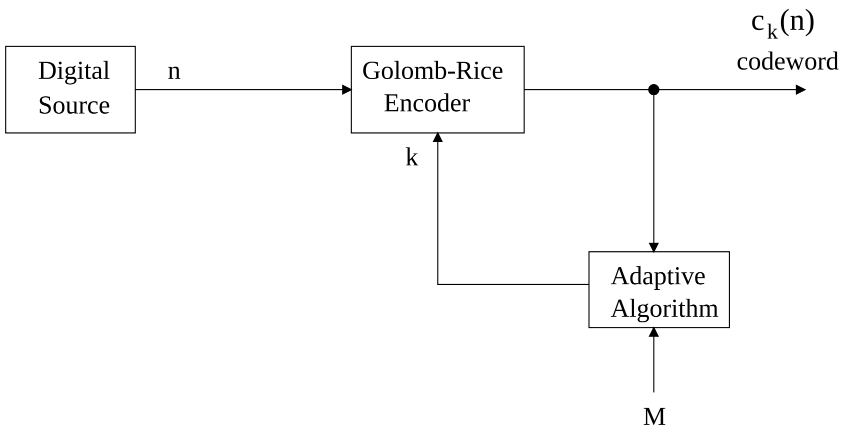


Figure 57: Feedback setup of the proposed algorithm.

The proposed algorithm relies on the following observation about the Golomb-Rice codes. Suppose that the integer n is encoded with a Golomb-Rice code with a given value of the parameter k. Then, the length of the codeword, $c_k(n)$, is $l_k = k + 1 + u_k$, where u_k is the number of ones in the corresponding unary code and is equal to $\lfloor n/2^k \rfloor$. If the code parameter k is incremented by one, the length of the new codeword would be $l_{k+1} = k + 2 + \lfloor u_k/2 \rfloor$. If the code parameter k is decremented by one, the resulting codeword length is $l_{k-1} = k + 2u_k + b_{k-1}$, where b_{k-1} is the k^{th} bit of the input n, i.e. $b_{k-1} = 0$ if $\lfloor n/2^{k-1} \rfloor$ is even, and $b_{k-1} = 1$ otherwise. Table 3 shows the codeword lengths for different values of u_k.

Table 3: Codeword lengths

	l_{k-1}	l_k	l_{k+1}
$u_k = 0$	$k + b_{k-1}$	$k + 1$	$k + 2$
$u_k = 1$	$k + 2 + b_{k-1}$	$k + 2$	$k + 2$
$u_k = 2$	$k + 4 + b_{k-1}$	$k + 3$	$k + 3$
$u_k > 2$	$\geq k + 6 + b_{k-1}$	$\geq k + 4$	$\geq k + 3$

From Table 3 notice that when $u_k = 0$ and $b_{k-1} = 0$, a 1-bit saving in the codeword length is achieved if the code parameter k is decremented by one. Similarly, if $u_k > 2$ a 1-bit saving is obtained if the code parameter k is incremented by one. This observation provides a simple way to update k based on the last output of the encoder. However, to better capture the average behavior of the source, previous outputs of the encoder should also be taken into account. In the proposed algorithm this is achieved by counting the number of times the conditions to increment or decrement the value of k are met. If this count reaches a certain threshold M, the value of k is then updated. The following pseudo-code summarizes the algorithm. Notice that in the algorithm the condition to increment k, that is $u_k > 2$, has been replaced with $u_k > 1$. It has been observed from simulations that this change results in a faster adaptation of the encoder.

```
1) Initialize $k$ to $k_{ini}$;
2) Reset counter;
3) Read input $n$ and encode it using $k$,
   output codeword $c_k(n)$;
4) If ($u_k > 1$) increment counter;
5) If ($u_k = 0$ AND $b_{k-1} = 0$) decrement counter;
6) If (counter value $\geq M$) k=k+1; Goto 2;
7) If (counter value $\leq -M$) k=k-1; Goto 2;
8) Goto 3;
```

The proposed algorithm is not especially sensitive to the initial value of k, k_{ini}. Typically k_{ini} would be set to the middle of the range of k. Large values of M will better capture the average behavior of the source but will also slow down the transient response of the algorithm. On the other hand, small values of M will speed up its response to changes in the source statistics but if the source statistics change rapidly, k will not converge, it will oscillate and generate unnecessary bits. It has been observed through extensive simulations with natural images that a value of $M = 4$, provides good compression performance in both cases.

A similar strategy known as the block-Melcode algorithm first described in [61] is used in the run-length mode in LOCO-I/JPEG-LS. In this strategy a run of length $m = 2^g$ increments an index, otherwise the index is decremented. The index is used to access a table which determines when g will be incremented or decremented [45].

The proposed algorithm requires an up/down counter and some logic for its implementation. It does not require accumulators or data buffers and, despite its simplicity, performs remarkably well. Figure 58 shows a schematic diagram that implements the algorithm. The inputs d_1 and d_2 are the first and second bits of the unary code. The Glue Logic block detects the conditions for incrementing or decrementing the counter (steps 4 and 5 of the algorithm) and enables the counter's clock signal.

The size of the up/down counter is $\lceil \log_2 2M \rceil$ bits. For the case of $M = 4$, a 3-bit up/down digital counter is sufficient. In practical implementations the length of the unary code has to be limited to a finite value. In this work the maximum length of the unary code has been set to eight. If for a particular symbol the length of the unary code equals or exceeds eight, an uncompressed binary word for the input symbol is appended to the eight-bit long unary code. Thus, an eight-bit long unary code works as an escape symbol.

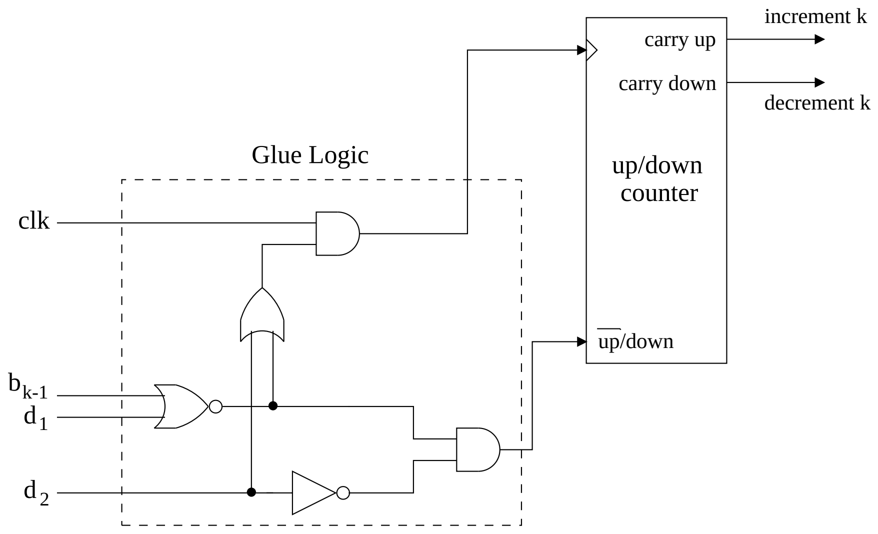


Figure 58: Schematic diagram of the proposed adaptive algorithm.

A synchronous 3-bit up/down counter can be implemented with three T flip-flops, 1 AND gate, and 2 XOR gates [60]. Additional to this, a circuit that resets the counter when it reaches its limits is needed which gives a complexity of 114 CMOS transistors just for the counter. Adding to this complexity measure the number of transistors needed to implement the Glue Logic block of Figure 58, the final complexity measure of the proposed encoder is 138 CMOS transistors.

Results

An initial evaluation of the performance of the proposed low-complexity adaptive algorithm has been done with a stationary one-sided geometric distribution (OSGD) and a stationary two-sided geometric distribution (TSGD). The distribution function for an OSGD source is given by

$$P(n) = (1 - \theta)\theta^n \qquad 0 < \theta < 1 \tag{51}$$

where n is a non-negative integer. The distribution function for a TSGD source is given by

$$P(x) = \frac{1-\theta}{\theta^{1-d}+\theta^{d}}\theta^{|x+d|} \qquad x = 0, \pm 1, \pm 2, \cdots \qquad 0 < \theta < 1 \tag{52}$$

with $0 \leq d \leq 1/2$. The parameter d determines the offset of the distribution, $d = 0$ corresponds to a centered distribution. The integer mapping of Equation (46) is applied to the output of the TSGD source to provide the non-negative integers required by encoder.

Figures 59 and 60 show the redundancy in bits/symbol of the proposed algorithm along with the redundancies of the LOCO and Rice algorithms for the OSGD and TSGD sources, respectively. The average codeword length for each one of the algorithms was computed by generating and encoding 10^5 random variables for each value of θ. The N_o parameter of LOCO was set to 64 and in all cases the dynamic range of k is from 0 to k_m. The parameter k_m is the maximum value that k is allowed to take on. It is set to 7 yielding 8 coding options in the Rice algorithm. Therefore, the length of the ID identifier each block of $J = 16$ symbols is 3 bits. In the simulations the parameter d of the TSGD source is set to 0.

For the OSGD case, it can be seen from the figure that the proposed low-complexity adaptive algorithm performs better than the Rice algorithm and for values of $\theta < 0.65$ it closely follows the LOCO algorithm. For the TSGD case, the proposed algorithm has lower redundancy than the Rice machine but the LOCO adaptation rule performs better than both of them. FELICS, however, seems to have the best performance.

Mathematical Analysis

The behavior of the up/down counter is modeled with a Markov chain. Each state in the Markov chain corresponds an state of the counter. Figure 61 shows the state diagram for the Markov chain. The states are denoted $s_{k,i}$, where $0 \leq k \leq k_m$ and $-M \leq i \leq M$. The index k denotes the current value of the Golomb-Rice parameter and the index i denotes the current value of the up/down counter. Recall that when the counter reaches one of its counting limits (M or $-M$), the counter is reset to 0 and k is updated. Thus, there are branches with probability 1 going from state $s_{k,M}$ to $s_{k+1,0}$ and from $s_{k,-M}$ to $s_{k-1,0}$. Similarly, and since k is restricted to be less than or equal to k_m, the probability of going from state $s_{k_m,M}$ to state $s_{k_m,0}$ is set to 1. All other transition probabilities are equal to 0. The total number of states is $N_s = (2M+1)(k_m+1)$.

The probability of counting up or equivalently of going from state $s_{k,i}$ to $s_{k,i+1}$ is

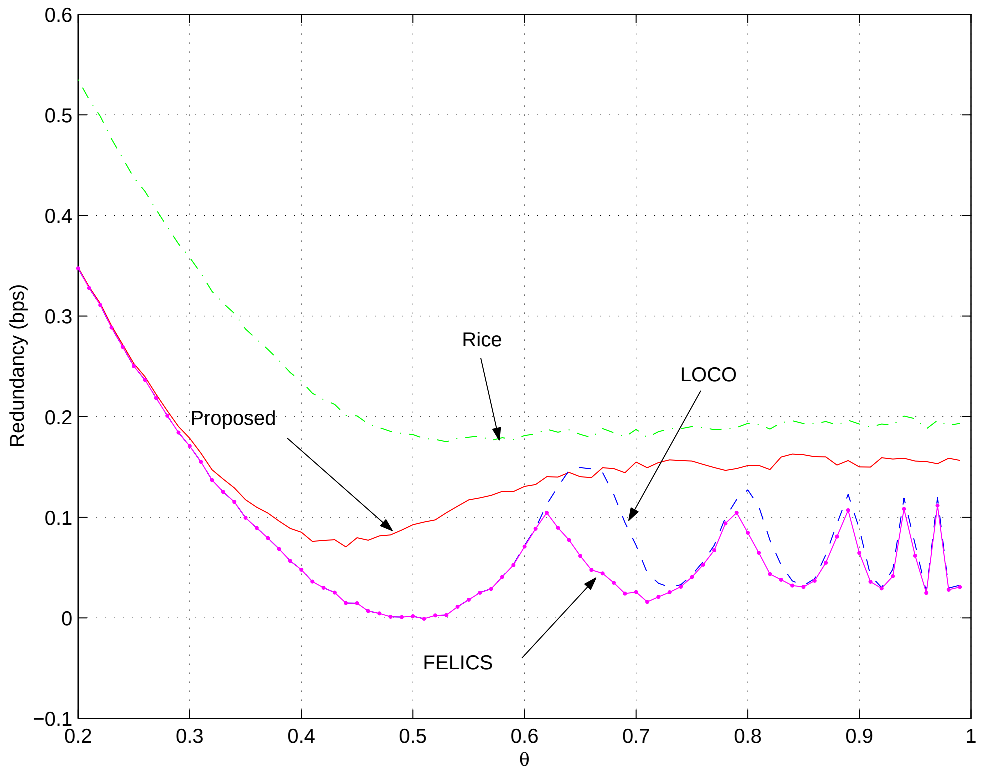


Figure 59: Redundancy of the proposed, LOCO, Rice, and FELICS algorithms for an OSGD source.

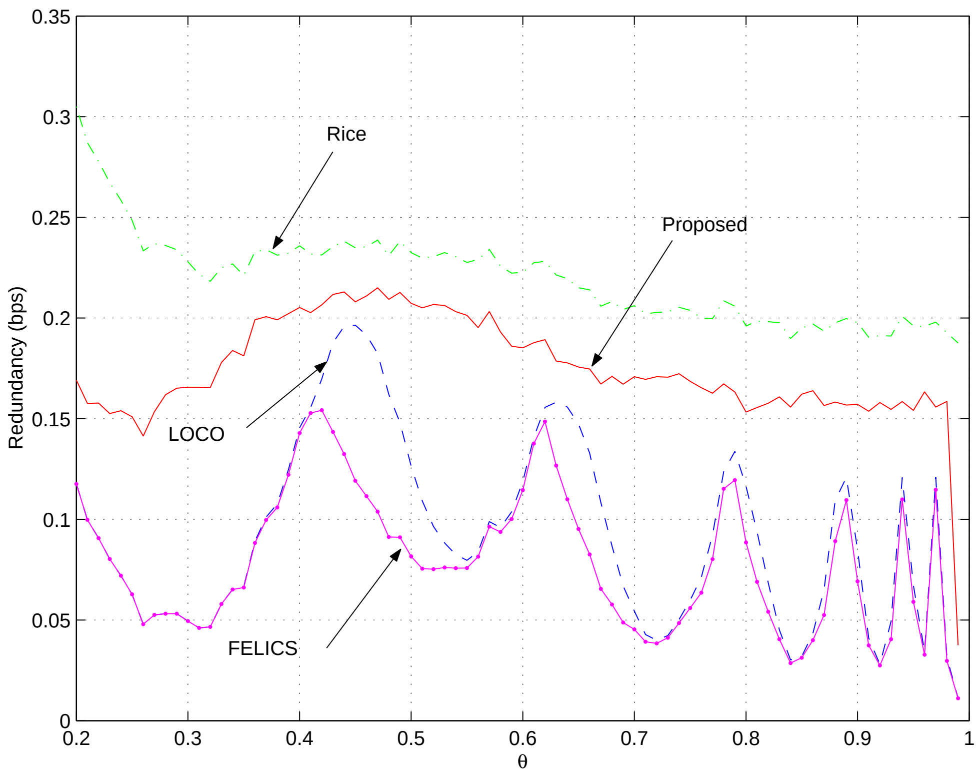


Figure 60: Redundancy of the proposed, LOCO, Rice, and FELICS algorithms for a TSGD source.

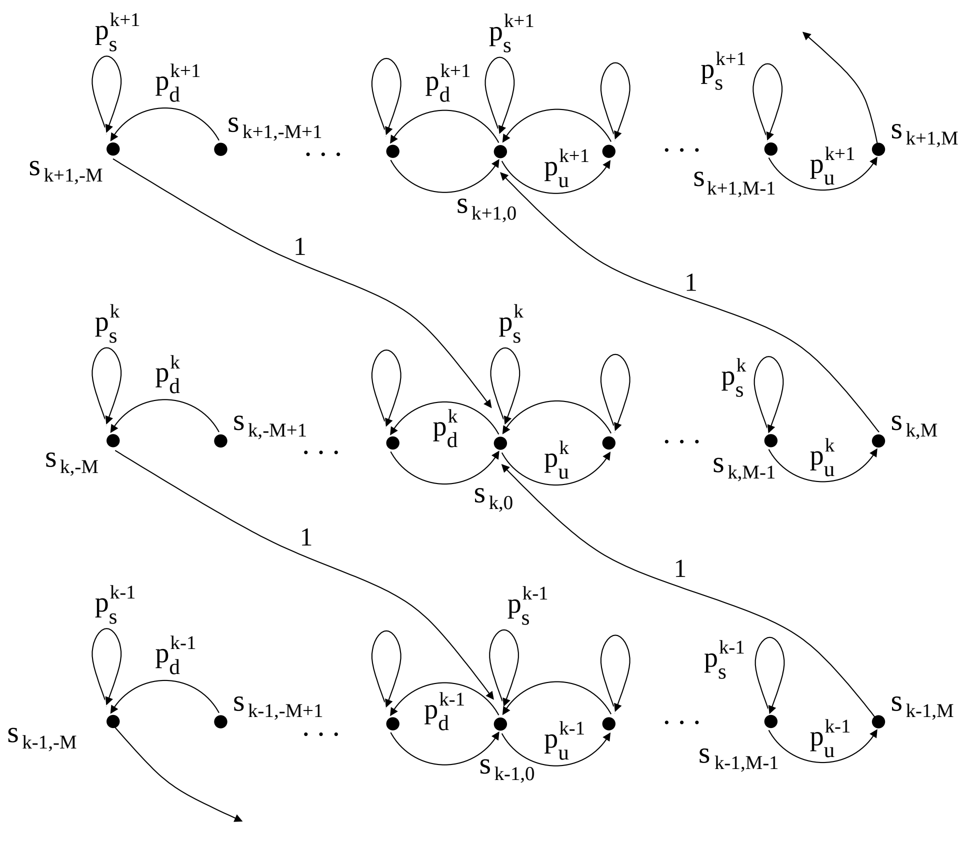


Figure 61: Markov chain modeling the operation of the up/down counter.

p_u^k. Given a value of k and assuming an OSGD source this probability is given by

$$p_u^k = \Pr\{u_k > 1\} = \Pr\{n \geq 2^{k+1}\} = \sum_{j \geq 2^{k+1}}^{\infty} (1-\theta)\theta^j = \theta^{2^{k+1}} \tag{53}$$

The probability of counting down or equivalently going from state $s_{k,i}$ to $s_{k,i-1}$ is p_d^k. For an OSGD source this is given by

$$\begin{aligned} p_d^k &= \Pr\{u_k = 0 \ \& \ b_{k-1} = 0\} = \Pr\{n < 2^{k-1}\} = \\ &= \sum_{j=0}^{2^{k-1}-1} (1-\theta)\theta^j = 1 - \theta^{2^{k-1}} \qquad\qquad k > 0 \end{aligned} \tag{54}$$

For $k = 0$, $p_d^k = \Pr\{u_k = 0\} = \Pr\{n = 0\} = 1 - \theta$. The probability that the counter does not count up nor does count down is denoted by $p_s^k = 1 - p_u^k - p_d^k$.

For the TSGD case the probabilities of counting up and down are given by

$$p_u^k = \Pr\{u_k > 1\} = \Pr\{n \geq 2^{k+1}\} = \theta^{2^k} \tag{55}$$

and

$$p_d^k = \begin{cases} 1 - \theta^{2^{k-2}} & \text{if } k > 2 \\ 1 - \theta & \text{if } k = 2 \\ \frac{1-\theta}{\theta^{1-d} + \theta^d}\theta^d & \text{if } k = 0, 1 \end{cases} \tag{56}$$

Recall that for obtaining the TSGD probabilities, the mapping of Equation (46) is applied before the encoding.

With these probability values the probability transition matrix $\mathbf{P}$ can be constructed. Given that this Markov chain is aperiodic and irreducible, the unique limiting state probabilities vector $\pi = [\pi_0, \ \pi_1, \ldots, \pi_{N_s-1}]$ can be computed by solving the following system of linear equations:

$$\pi = \pi\mathbf{P} \qquad\qquad \sum_{j=0}^{N_s-1} \pi_j = 1 \tag{57}$$

The probability q_k that the algorithm selects a given value of the parameter k can be obtained from the limiting state probability vector π by summing up the limiting probabilities that correspond to the states $s_{k,i}$, with $-M \leq i \leq M$, i.e.

$$q_k = \sum_{j=0}^{2M} \pi_{(2M+1)k+j} \tag{58}$$

The average codeword length is then given by

$$\bar{L} = \sum_{k=0}^{k_m} \sum_{n=-\infty}^{\infty} q_k p(n|\theta) l_{n,k} \tag{59}$$

where $p(n|\theta)$ is the probability that the source outputs the integer n given the value of θ. For the OSGD source, this probability is given by Equation (51) and for the TSGD source it is given by Equation (52). Finally $l_{n,k}$ is the length of the Golomb-Rice codeword for n and it is given by Equation (45). Figure 62 shows the theoretical and simulated redundancies in bits per sample (bps) achieved by the algorithm for an OSGD source while Figure 63 shows the redundancies for the TSGD case.

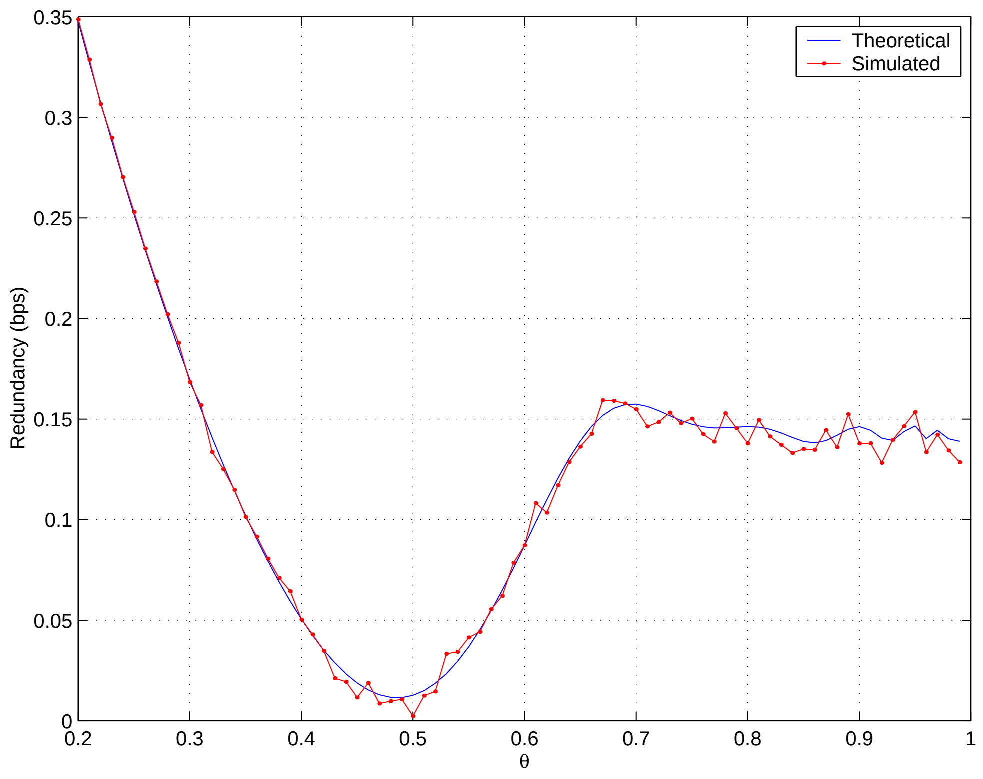


Figure 62: Theoretical and simulated redundancies of the proposed algorithm for an OSGD source.

The results in Figures 62 and 63 were obtained with $M = 4$. Figure 64 shows the algorithm performance for different values of M. For stationary sources a larger value of M results in a better compression performance. Additionally, Figure 65 shows the proposed algorithm performance with the performance of the LOCO algorithm [10].

Test with Natural Images

Finally, the proposed adaptive algorithm has been tested on a set of natural

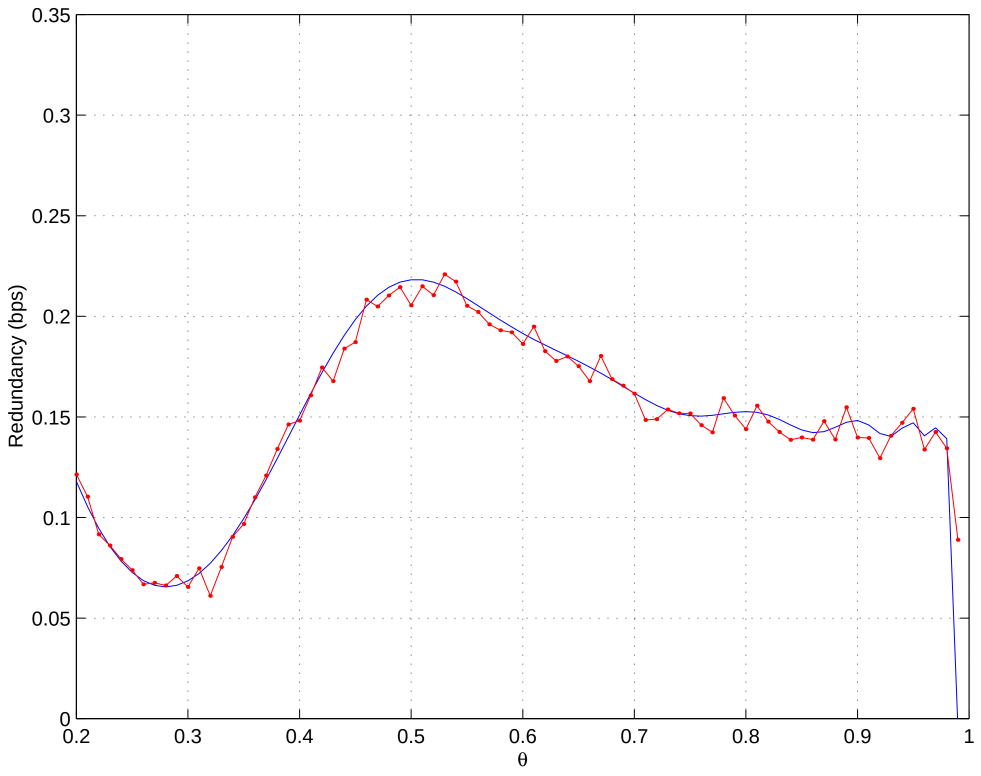


Figure 63: Theoretical and simulated redundancies of the proposed algorithm for a TSGD source.

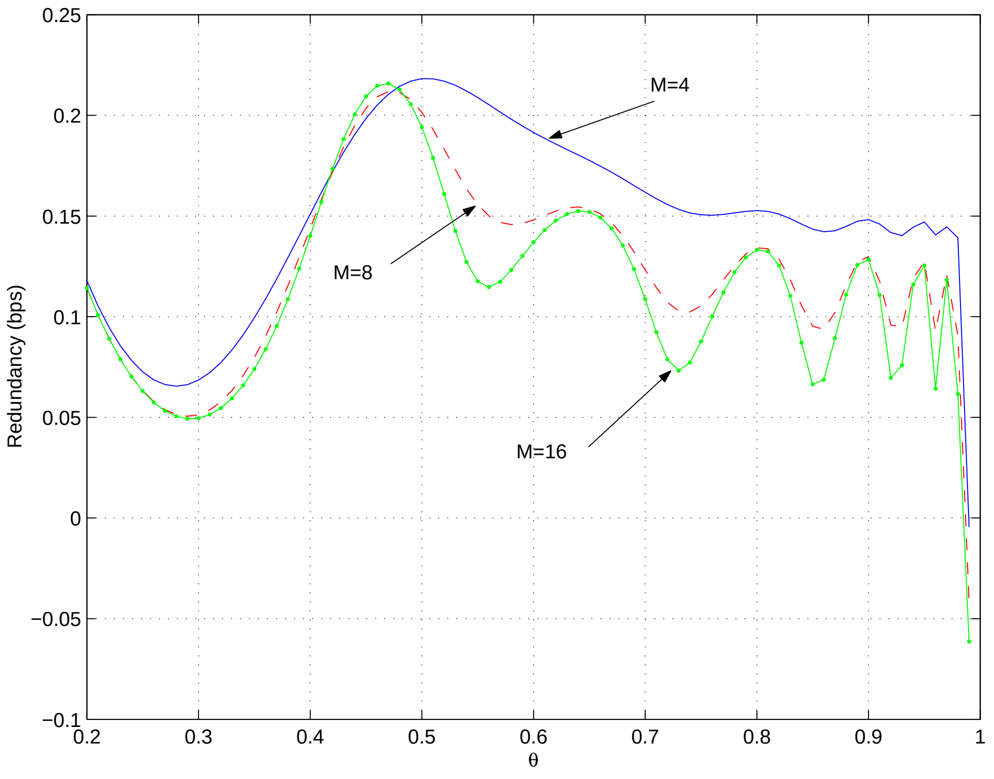


Figure 64: Theoretical redundancies for different values of the threshold M

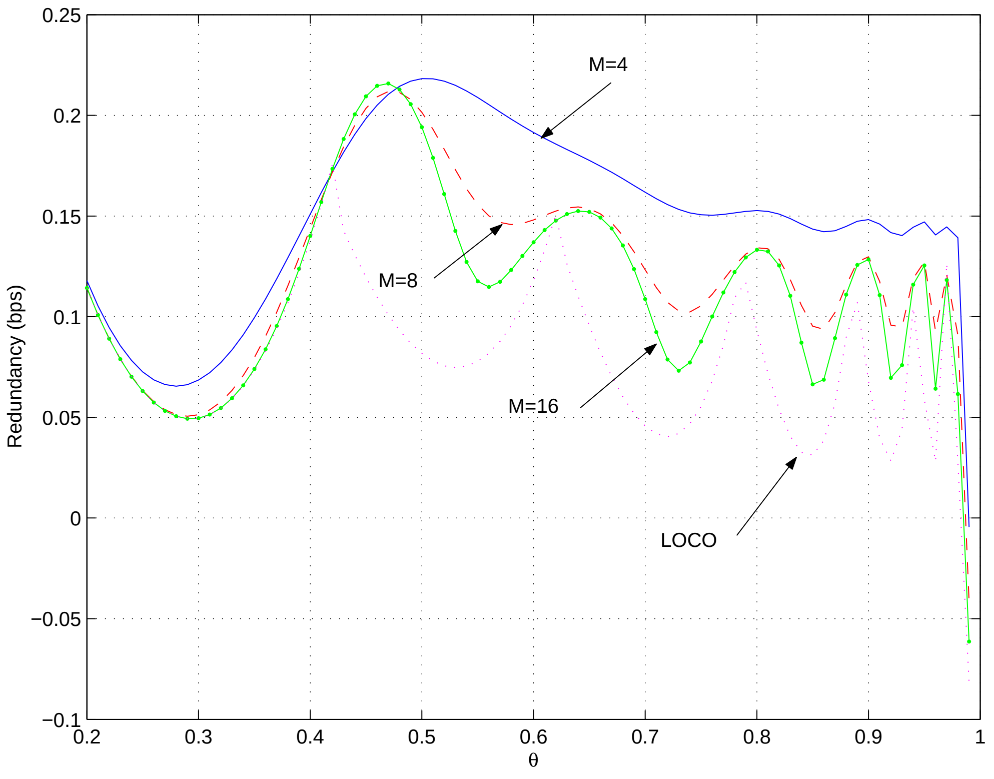


Figure 65: Theoretical redundancies for the proposed and LOCO adaptive algorithms

grayscale images. Inter-pixel correlation is first removed using the simplified gradient pixel prediction rule suggested [10]. The prediction residual, that is the difference between the actual pixel and its prediction, is encoded with a Golomb-Rice code. Since the prediction residuals can be negative, the mapping of Equation (46) is applied to the residuals before encoding. Table 4 shows the average codeword length of the residual image for the LOCO, Rice, and the proposed adaptive algorithms. The first two columns of the table show the first order entropy of the original and the residual images.

Table 4: Rate performance of the LOCO, Rice and proposed algorithms with natural images

File	Original	Residual	LOCO	FELICS	Rice	Proposed
sensin	7.3179	3.9481	3.8844	3.9220	3.4127	4.0182
mandril	7.2283	5.8908	5.8391	5.8811	5.8602	5.9543
peppers	7.5421	4.9111	4.9338	4.9442	4.9973	5.0601
raffia	6.8198	5.1787	5.2180	5.2160	5.2638	5.4309
mri	6.7863	4.8016	4.7381	4.8082	4.5907	4.9261
lena	7.4451	4.5585	4.5265	4.5544	4.4721	4.6330
aerial	6.9940	5.3515	5.3136	5.3717	5.0269	5.8730
wood	6.3562	4.4558	4.4959	4.5003	4.5912	4.6541
gravel	6.8087	4.8322	4.8699	4.8745	4.9363	5.0207
einstein	6.8841	4.8399	4.7763	4.8144	4.7437	4.9290
camera man	6.9046	5.1773	5.1535	5.2046	5.0206	5.5202
brain	6.7808	5.6098	5.5775	5.6905	5.4277	5.8175
goldhill	7.4778	4.8835	4.8189	4.7867	4.8159	4.9581
tiffany	6.5558	4.3281	4.2505	4.2693	3.9391	4.4087
drop	7.2532	4.3005	4.3875	4.4254	3.6732	4.6678

It can be noted from this table that the performance of the proposed adaptive algorithm is comparable to the performance of the other algorithms. The performance of the proposed algorithm can be enhanced if additional conditions for incrementing the counter's value are included. For instance, when the following conditions are added, the compression rate improves as shown in Table 5:

```
If ($u_k > 2$) increment counter;
If ($u_k > 3$) increment counter;
If ($u_k > 4$) increment counter;
If ($u_k > 5$) increment counter;
If ($u_k > 6$) increment counter;
```

The average improvement for the 15 images of Table 4 is 0.2035 bps. However, adding these extra conditions to the algorithm increases the hardware complexity.

A total of 80 extra CMOS transistors would be needed to implement these new conditions, which gives a grand total of 218 CMOS transistors for the enhanced adaptive algorithm.

Table 5: Rate performance of the LOCO, Rice and enhanced proposed algorithms with natural images

File	Original	Residual	LOCO	FELICS	Rice	Proposed
sensin	7.3179	3.9481	3.8844	3.9220	3.4127	3.8627
mandril	7.2283	5.8908	5.8391	5.8811	5.8602	5.8001
peppers	7.5421	4.9111	4.9338	4.9442	4.9973	4.9607
raffia	6.8198	5.1787	5.2180	5.2160	5.2638	5.2671
mri	6.7863	4.8016	4.7381	4.8082	4.5907	4.6992
lena	7.4451	4.5585	4.5265	4.5544	4.4721	4.4888
aerial	6.9940	5.3515	5.3136	5.3717	5.0269	5.3844
wood	6.3562	4.4558	4.4959	4.5003	4.5912	4.5456
gravel	6.8087	4.8322	4.8699	4.8745	4.9363	4.9090
einstein	6.8841	4.8399	4.7763	4.8144	4.7437	4.7676
camera man	6.9046	5.1773	5.1535	5.2046	5.0206	5.1965
brain	6.7808	5.6098	5.5775	5.6905	5.4277	5.4666
goldhill	7.4778	4.8835	4.8189	4.7867	4.8159	4.8507
tiffany	6.5558	4.3281	4.2505	4.2693	3.9391	4.2396
drop	7.2532	4.3005	4.3875	4.4254	3.6732	4.3812

6 Summary and Future Work

This research wok addressed the problem of focal plane video compression. The problem was tackled following a top-down approach. First, the problem was studied at the system level. It was immediately clear that standard image and video compression solutions were not suitable for focal plane integration due to their large complexities. Also existing image compression systems did not take advantage of the analog nature of the data at the sensor level and in their majority were designed to be implemented with a microprocessor. The approach to video compression taken in this research work was to implement a predictive coding in the analog domain. Predictive coding was chosen due to its lossless compression feature and its simplicity. Analog circuits were employed to avoid the quantization noise in the prediction loop and because many arithmetic functions can be implemented with small analog circuits.

At the system level we also identified a potential for great complexity savings if the A/D is merged with an entropy encoder. Such integration is possible by noticing that some A/D architectures like the single-slope converter have common circuitry with a Golomb-Rice entropy encoder. The result was a circuit that performs the analog-to-digital conversion and the entropy coding simultaneously while providing complexity savings. The impact of this integration on our research work was that for the first time the entropy coder could be implemented on the focal plane. In other approaches reported in the literature the entropy coding was implemented outside the focal plane usually by a computer. But by performing a joint quantization/coding we were able to implement the entropy coder not only on the sensor chip but at the column level.

Our research efforts did not stop at the system level. We went all the way down to a transistor level implementation. The result is a chip that contains the sensor, prediction circuits, A/Ds, entropy coders and read-out circuitry and provides at its output a compressed bit stream. The sensor is composed of an 80×44 array of active pixels. For each pixel column there is a corresponding mixed signal processor performing the tasks of prediction, analog-to-digital conversion, entropy coding and read-out and control circuitry. The final byproduct of this research work is a single chip solution that is able to acquire images and compress them providing at its output a compressed bit stream. The chip was implemented in a 0.35 μm 4-metal 2-poly CMOS technology.

Tests of the imager chip indicate that performing the pixel prediction in the analog domain introduced a number of extra sources of noise. One of them being FPN caused by the variations of the fabrication process parameters across the chip. CDS circuits were added at the column level to cancel out the effects of these variations. While they succeeded in removing they introduced FPN themselves despite precautions taken at the layout level. However, this new FPN can be compensated off-line. Another limitation of the analog implementation was evident during inter-frame prediction. The previous frame is stored in pixel-level capacitors. Again due to area constraints these capacitors were small in size. As it turns out their charge leaked significantly in the time the next frame was acquired. The result is an incomplete reconstruction of the video frames.

In conclusion, the analog circuits employed offer complexity and SNR advantages but do not provide very accurate results. Greater computation accuracy can be achieved in the analog domain but more complex circuits would have to be used. In areas of machine vision where image quality is not the driving force analog implementations have proved to be a good solution. Future work in the area of focal plane compression can benefit from the lessons learned and ideas developed here. Specifically, future implementation might consider:

- A chip level architecture with a single A/D. This will reduce power, area and noise. A faster A/D architecture like the cyclic converter proposed in Section 4 may suffice.
- Perform the CDS and the prediction computations with more accurate analog techniques like switched capacitor circuits.
- Explore image decorrelation techniques like wavelet decomposition. Wavelet decomposition promises to provide better compression ratios than predictive coding.

The results of this dissertation reach beyond the boundaries of the original problem of focal plane video compression. Joint adaptive quantization/coding can be of good use in other applications where low-complexity compression techniques are necessary. An example of a field with these needs is found in sensor networks. Future focal plane video compression endeavors may also benefit from the results presented here.

References

[1] M. Schanz, W. Brockherde, R. Hauschild, B. J. Hosticka, and M. Schwarz, "Smart CMOS image sensor arrays," *IEEE Trans. Electron Devices*, vol. 44, no. 10, pp. 1699-1705, Oct. 1997.

[2] R. Forchheimer, K. Chen, C. Svensson, and A. Ödmark, "Single-chip image sensors with a digital processor array," *Journal of VLSI Signal Processing*, vol. 5, pp. 121-131, 1993.

[3] K. Aizawa, T. Hamamoto, Y. Ohtsuka, M. Hatori, M. Abe, "Implementations of on-sensor image compression and comparisons between pixel and column parallel architectures," *International Conference on Image Processing*, vol. 2, pp. 258-261, Oct. 1997.

[4] S. Kawahito, M. Yoshida, M. Sasaki, K. Umehara, D. Miyazaki, Y. Tadokoro, K. Murata, S. Doushou, and A. Matsuzawa, "A CMOS image sensor with analog two-dimensional DCT-based compression circuits for one-chip cameras," *IEEE J. Solid-State Circ.*, vol. 32, no. 12, pp. 2030-2041, 1997.

[5] S. E. Kemeny, H. H. Torbey, H. E. Meadows, R. A. Bredthauer, M. A. La Shell, and E. R. Fossum, "CCD focal-plane image reorganization processors for lossless image compression," *IEEE J. Solid-State Circ.*, vol. 27, no. 3, pp. 398-405, 1992.

[6] Z. Zhou, B. Pain, and E. R. Fossum, "CMOS active pixel sensor with on-chip successive approximation analog-to-digital converter," *IEEE Trans. Electron Devices*, vol. 44, no. 10, pp. 1759-1763, 1997.

[7] G. Torelli, L. Gonzo, M. Gottardi, F. Maloberti, A. Sartori, and A. Simoni, "Analog-to-digital conversion architectures for intelligent optical sensor arrays," *SPIE Proc.*, vol. 2950, pp. 254-264, 1996.

[8] P. M. Acosta-Serafini, I. Masaki, C. G. Sodini, "A 1/3$''$ VGA linear wide dynamic range CMOS image sensor implementing a predictive multiple sampling algorithm with overlapping integration intervals," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 9, pp. 1487-1496, Sept. 2004.

[9] A. Moini, *Vision Chips*, Kluwer Academic Publishers, Boston, 2000.

[10] M. J. Weinberger, G. Seroussi, and Guillermo Sapiro, "LOCO-I: a low complexity, context-based, lossless image compression algorithm," *Data Compression Conference*, pp. 140-149, 1996.

[11] W. D. Leon-Salas, S. Balkir, K. Sayood, and M. W. Hoffman, "An analog-to-digital converter with Golomb-Rice output codes," *IEEE Transactions on Circuits and Systems-II*, vol. 53, no. 4, pp.278-282, April 2006.

[12] A. Fish, D. Turchin, and O. Yadid-Pecht, "An APS with 2-D WTA selection employing adaptive spatial filtering, bad pixel elimination and false alarm reduction," *IEEE International Symposium on Circuits and Systmes*, vol. 2, pp. 328-331, May 2002.

[13] A. Gopalan and R. R. Harrison, “A CMOS imager with on-chip temporal filtering for motion pre-processing,”*IEEE International Symposium on Circuits and Systmes*, vol. 2, pp. 336-339, May 2002.

[14] S. Mehta and R. Etienne-Cummings, “A simplified normal optical flow measurement CMOS camera,” *IEEE Transactions on Circuits and Systems-I*, vol. 53, no. 6, pp. 1223-1234, June 2006.

[15] C. P. Chong, C. A. T. Salama, and K. C. Smith, “Image-motion detection using analog VLSI,” *IEEE Journal of Sloid-State Circuits*, vol. 32, no. 1, pp. 93-96, Jan. 1992.

[16] A. Simoni, G. Torelli, F. Maloberti, A. Sartori, S. E. Plevridis, and A. N. Birbas, “A single-chip optical sensor with analog memory for motion detection,” *IEEE Journal of Solid-State Circuits*, vol. 30, no. 7, pp. 800-805, July 1995.

[17] K. Nishio, H. Yonezu, S. Sawa, Y. Yoshikawa, and Y. Furukawa, “Analog integrated circuit for detection of approaching object against moving background based on lower animal vision,” *IEEE International Symposium on Circuits and Systmes*, vol. 4, pp. 832-835, May 2004.

[18] R. Etienne-Cummings, P. Pouliquen, and M. A. Lewis, “Color segmentation, histogramming and pattern matching chip,” *Electronic Letters*, vol. 38, no. 4, pp. 172-174, Feb. 2002.

[19] A. Pesavento, C. Koch, “A CMOS imager with focal-plane computation for feature detection,” *IEEE International Symposium on Circuits and Systmes*, vol. 3, pp. 624-627, May 2001.

[20] R. Dominguez-Castro, S. Espejo, A. Rodriguez-Vazquez, R. Carmona, P. Foldesy, A. Zarandy, T. Sziranyi, and T. Roska, “A 0.8-μm CMOS two-dimensional programmable mixed-signal focal-plane array processor with on-chip binary imaging and instructions storage,” *IEEE Journal of Solid-State Circuits*, vol. 32, no. 7, pp. 1013-1025, July 1997.

[21] S. Espejo, A. Rodriguez-Vazquez, R. Dominguez-Castro, J. L. Huertas, and E. Sanchez-Sinencio, “Smart-pixel cellular neural networks in analog current-mode CMOS technology,” *IEEE Journal of Solid-State Circuits*, vol. 29, no. 8, pp. 895-905, Aug. 1994.

[22] K. Sayood, *Introduction to Data Compression, 2nd ed.*, Academic Press, San Francisco, 2000.

[23] K. Sayood, "Analysis of differential PCM," *Asilomar IEEE Conference on Circuits, Systems, and Computers*, vol. 1, pp. 218-222, 1985.

[24] H. A. Spang III and P. M. Schulthesis, "Reduction of quantization noise by use of feedback," *IEEE Transactions on Communications and Systems*, vol. CS-10, pp. 373-380, Dec. 1982.

[25] K. Sayood and J. D. Gibson, "Explicit additive noise models for uniform and nonuniform MMSE quantization," *Signal Processing*, vol. 7, pp. 407-414, 1984.

[26] S. Kawahito, D. Handoko, Y. Tadokoro, "A CMOS image sensor with motion vector estimator for low-power image compression," *Instrumentation and Measurement Technology Conference*, vol. 1, pp. 65-70, May 1999.

[27] K. Iizuka, M. Miyamoto, H. Matsui, and K. Hashiguchi, "A 0.2b/pixel 16mW real-time analog image encoder in 0.8μm CMOS," *International Solid-State Circuits Conference*, pp. 190-191, 154, Feb. 1997.

[28] D. A. Johns and K. Martin, *Analog Integrated Circuit Design*, John Wiley & Sons, New York, 1997.

[29] J. Chang, A. A. Abidi, and C. R. Viswanathan, "Flicker noise in CMOS transistors from subthreshold to strong inversion at various temperatures," *IEEE Transactions on Electron Devices*, vol. 41, no. 11, pp. 1965-1971, Nov. 1994.

[30] J. Zhou, M. Cheng, and L. Forbes, "SPICE models for flicker noise in p-MOSFETs in the saturation region," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 6, pp. 763-767, June 2001.

[31] D. Xie, M. Cheng, and L. Forbes, "SPICE models for flicker noise in n-MOSFETs from subthreshold to strong inversion," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 11, pp. 1293-1303, Nov. 2000.

[32] J. McCreary and P. R. Gray, "All-MOS charge redistribution analog-to-digital conversion techniques-Part 1," *IEER Journal of Solid-State Circuits*, vol. SC-10, pp. 371-379, Dec. 1975.

[33] Mark G. Johnson, "An input-free V_T extractor circuit using a two-transistor differential amplifier," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 6, pp. 704-705, June 1993.

[34] Xiang Liang Jin and Jie Chen, "Novel principle and realization of simple low-power low-noise correlated double sampling for CMOS pixel readout circuit," pp. 113-116, 2003.

[35] R. Peck and D. Schröeder, "A low-power entropy-coding analog/digital converter with integrated data compression," *European Solid-State Conf.*, pp. 173-176, Sept. 2003.

[36] D. Martinez, "Time-adaptive vector A/D conversion," *IEEE Trans. Circuits and Systems-II*, vol. 45, no. 10, pp. 1420-1424, Oct. 1998.

[37] R. F. Rice, "Some practical universal noiseless coding techniques," *JPL Publication 79-22*, Jet Propulsion Laboratory, Pasadena, California, Mar. 1979.

[38] A. Gersho, "Principles of quantization," *IEEE Trans. Circuits and Syst.*, vol. CAS-25, no. 7, pp.427-436, July 1978.

[39] R. H. McCharles, V. A. Saletore, W. C. Black, and D. A. Hodges, "An algorithmic analog-to-digital converter," *IEEE International Solid-State Circuits Conference*, vol. XX, pp. 96-97, Feb. 1977.

[40] H. Ohara, H. X. Ngo, M. J. Armstrong, C. F. Rahim, and P. R. Gray, "A CMOS programmable self-calibrating 13-bit eight-channel data acquisition peripheral," *IEEE Journal of Solid-State Circuits*, vol. SC-22, no. 6, pp. 930-938, Dec. 1987.

[41] H-S. Lee, "A 12-b 600 ks/s digitally self-calibrated pipelined algorithmic ADC," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 4, pp. 509-515, April 1994.

[42] O. E. Erdoğan, P. J. Hurst, and S. H. Lewis, "A 12-b digital-background-calibrated algorithmic ADC with -90-dB THD," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 12, pp. 1812-1820, Dec. 1999.

[43] S. Golomb, "Run-length encodings," *IEEE Trans. Information Theory*, vol. 12, no. 3, pp. 399-401, 1966.

[44] R. G. Gallager and D. C. Van Voorhis, "Optimal source codes for geometrically distributed integer alphabets," *IEEE Trans. Information Theory*, vol. 21, no. 2, pp. 228-230, 1975.

[45] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: principles and standarization into JPEG-LS," *IEEE Trans. on Image Processing*, vol. 9, no. 8, pp. 1309-1324, Aug. 2000.

[46] N. Merhav, G. Seroussi, and M. J. Weinberger, "Coding of sources with two-sided geometric distributions and unknown parameters," *IEEE Transactions on Information Theory*, vol. 46, no. 1, pp. 229-236, Jan. 2000.

[47] K. Sayood, *Lossless Compression Handbook*, Academic Press:San Diego, 2003.

[48] R. F. Rice, "Lossless coding standards for space data systems," *Asilomar Conf. Signals, Syst. and Computers*, vol. 1, pp. 577-585, Nov. 1996.

[49] P. G. Howard and J. S. Vitter, "Fast and efficient lossless image compression," *Data Compression Conference*, pp. 351-360, Mar. 1993.

[50] T. Liebchen and Y. A. Reznik, "MPEG-4 ALS: an emerging standard for lossless audio coding," *Proceedings of the Data Compression Conference*, pp. 439-448, 2004.

[51] M. Hans and R. W. Schafer, "Lossless compression of digital audio," *IEEE Signal Processing Magazine*, vol. 18, no. 4, pp. 21-32, 2001.

[52] G. Seroussi and M. J. Weinberger, "On adaptive strategies for an extended family of Golomb-type codes," *Data Compression Conference*, pp. 131-140, 1997.

[53] T. Robinson, "SHORTEN: Simple lossless and near-lossless waveform compression," Technical Report TR.156, Cambridge Univ. Eng. Dept., Cambridge, UK, 1994.

[54] J. Venbrux, Pen-Shu Yeh, and M.N. Liu, "A VLSI chip set for high-speed lossless data compression," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 2, no. 4, pp. 381-391, Dec. 1992.

[55] R. F. Rice, "Some practical universal noiseless coding techniques–Parts I-III," Jet Propulsion Laboratory, Pasadena, CA, Tech. Rep. JPL-79-22, JPL-83-17, JPL-91-3, 1975, 1983, and 1991.

[56] P.-S. Yeh, R. F. Rice and W. H. Miller, "On the optimality of a universal noiseless coder," *Proc. AIAA Computing in Aerospace 9 Conference*, pp. 490-498, San Diego, Oct 1993.

[57] K. Cheung and P. Smyth, "A high speed distortionless predictive image compression scheme," *Proc. IEEE Symp. Inform. Theory and its Applic.*, pp. 467-470, 1990.

[58] Pen-Shu Yeh, "The CCSDS lossless data compression recommendation for space applications," *Lossless Compression Handbook*, K. Sayood, Editor, Academic Press, pp. 311-326, 2003.

[59] N. Merhav, G. Seroussi, and M. J. Weinberger, "Optimal prefix codes for sources with two-sided geometric distributions," *IEEE Trans. Information Theory*, vol. 46, no. 1, pp. 121-135, 2000.

[60] E. D. Fabricius, *Modern Digital Design and Switching Theory*, CRC Press, Boca Raton, Florida, 1992.

[61] R. Ohnishi, Y. Ueno, and F. Ono, "The efficient coding scheme for binary sources," *IECE Japan*, vol. 60-A, pp. 1114-1121, Dec. 1977. In Japanese.

[62] J. Doernberg, H.-S. Lee, and D. A. Hodges, "Full-speed testing of A/D converters," *IEEE Journal of Solid-State Circuits*, vol. SC-19, no. 6, pp. 820-827, Dec. 1984.

[63] F. Azaïs, S. Bernard, Y. Bertrand and M. Renovell, "Towards an ADC BIST scheme using the histogram test technique," *IEEE European Test Workshop*, pp. 53-58, May 2000.

[64] R. D. Yates and D. J. Goodman, *Probability and Stochastic Processes*, John Wiley & Sons, New York, 1999.

Focal Plane Video Compression

PUBLICATIONS

Refereed Journals

- W. León-Salas, N. Schemm, S. Balkir, K. Sayood, and M. Hoffman, "A CMOS image sensor with focal plane compression," in preparation for *IEEE Journal of Solid-State Circuits*.
- Z. Lin, Michael W. Hoffman, Nathan Schemm, Walter León-Salas, and Sina Balkir, "A CMOS image sensor for focal plane decomposition," being submitted to *IEEE Transactions on Circuits and Systems I*, December 2006.
- W. León-Salas, S. Balkir, K. Sayood, and M. Hoffman, "An adaptive Golomb-Rice entropy coder for on-sensor compression," being submitted to *IEEE Transactions on Circuits and Systems I*, December 2006.
- W. León-Salas, S. Balkir, K. Sayood, and M. Hoffman, "An analog-to-digital converter with Golomb-Rice output codes," *IEEE Transactions on Circuits and Systems II*, vol. 53, no. 4, pp. 278-282, April 2006.

Conferences

- W. León-Salas, S. Balkir, K. Sayood, M. Hoffman, and N. Schemm, "A CMOS imager with focal plane compression," *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 3570-3573, Kos Island, Greece, May 2006.
- Z. Lin, Michael W. Hoffman, Walter León-Salas, Nathan Schemm, and Sina Balkir, "Effects of charge-based computation non-idealities on CMOS image compression sensors'" *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 5187-5190, Kos Island, Greece, May 2006.
- W. León-Salas, S. Balkir, K. Sayood, H. Hoffman, "An analog-to-digital converter with Golomb-Rice output codes," *IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 6, pp. 5549-5552, Kobe, Japan, May 2005.
- Z. Lin, Michael W. Hoffman, Walter D. León-Salas, Nathan Schemm, and Sina Balkir, "A CMOS image sensor for focal plane decomposition," *IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 5, pp. 5322-5325, Kobe, Japan, May 2005.
- W. León, S. Balkir, K. Sayood, and M. Hoffman, "Charge-based prediction circuits for focal plane image compression," *IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 4, pp. 936-939, Vancouver, Canada, May 2004.
- W. León, S. Balkir, K. Sayood, and M. Hoffman, "A CMOS imager with pixel prediction for image compression," *IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 4, pp. 776-779, Bangkok, Thailand, May 2003.
- W. León, S. Balkir, K. Sayood, and M. Hoffman, "An evolvable predictor for lossless image compression *IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 4, pp. 731-734, Phoenix, Arizona, May 2002.