A 3.9 Compression-Ratio Huffman Encoding Scheme for the Large Ion Collider on 65nm and 130nm CMOS technologies

Edwin G. Carreño , Christian D. Hernandez, Óscar M. Diaz, Héctor Gómez, Carlos Fajardo, Hugo Hernandez[1]
Wilhelmus Van Noije[2] and Elkim Roa
Design Group of Integrated Systems CIDIC - Universidad Industrial de Santander - Bucaramanga, Colombia
[2]University of Sao Paulo, Sao Paulo, Brazil

***Abstract*—A Huffman coding scheme with 3.9 compression ratio for the Large Hadron Collider experiment is proposed. A fully-synthesized scheme draws a small footprint layout of $60\mu m \times 60\mu m$ in 65nm and $105\mu m \times 105\mu m$ in 130nm CMOS process. The maximum operation frequencies are 435MHz for 65nm and 333MHz for 130nm, whereas the power consumption is 1.2mW and 1.9mW respectively. The resulting scheme enables a front-end electronics without any loss of data.**

***Index Terms*—Data compression, Digital circuits, Integrated circuit synthesis, Huffman encoding.**

I. Introduction

At the heart of the Large Hadron Collider (LHC) at CERN (Conseil Europen pour la Recherche Nuclaire) there are thousands of particle sensors with a complex electronic system. Different chips are installed to handle the sensor signals and to communicate the data out to data-centers for processing. At the front-end, readout chips are used to amplify and filter signals within multiple-channels system-on-chips (SOC).

A common bottleneck in multiple channels front-end SOC is the data communication. Getting the data out for processing requires high-speed interfaces with high-efficiency that are able to cope with lossy channels. However, power consumption limitation in channels of the current LHC infrastructure imposes the need for data compression. Current working architecture of the front-end electronics does not include a compression scheme implying the possibility to lose data because of channel limitations [1].

This work proposes a compression scheme to increase the effective data-rate send out from the readout chips. The proposed scheme uses Huffman coding which is able to compress data at an average of 3.9 compression ratio. After a statistical data analysis we proved that a compact Huffman dictionary can be generated to compress data at reduced power consumption. The encoding scheme is implemented on $65nm$ and $130nm$ CMOS technologies and verified. Simulation results of fully-synthesized circuitry showed that the implementation paves the way to a complete-efficient front-end circuitry without lossing data.

[1]Hugo Hernandez is also with the ALICE project at CERN.

II. Front end - datapath collider

The A.L.I.C.E (A Large Ion Collider Experiment) project is one of four major experiments of particle accelerator LHC installed at CERN. An upgrade program has been approved for the management committee of LHC and the different updates are looking forward to improving the resolution and tracking efficiency maintaining particles identification ability. To achieve this, it is necessary to update the read out electronics in the Time Projection Chamber (TPC detection chamber) [1].

One of many updates involves a readout circuit called the SAMPA chip, this ASIC transforms the charge signal into a differential semi-Gaussian voltage signal. This signal is digitized by a 10-bit ADC with a sample ratio of 10MHz. After the ADC a digital signal processor eliminates any signal perturbations, distortion of the shape, offset and signal variation due to temperature variations. The SAMPA sends out the data using serial links after the DSP. Fig. 1 shows the front-end starting from DSP to Serial links.

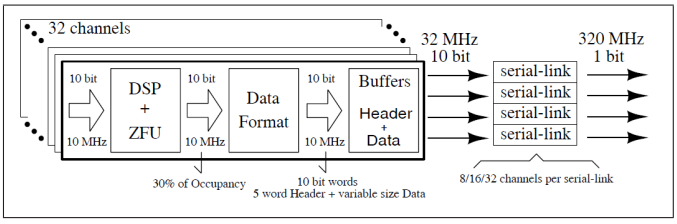


Fig. 1 ALICE TPC front end electronics [1].

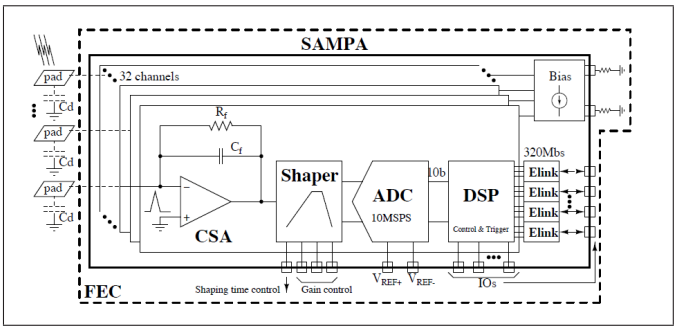


Fig. 2 SAMPA system block diagram [1].

ISBN 978-1-4673-7835-2/16/$31.00©2016 IEEE

IEEE Catalog Number CFP16LAS-ART

Fig. 2 shows the block diagram of SAMPA chip. The data compressor in this document will be between DSP Output and E-Links (serial links).

III. Huffman architecture

A. Lossless Huffman compression algorithm

The Huffman coding is a prefix code that is used for lossless data compression. This algorithm is optimal because the length of the compressed data (L) is not greater than 1 bit of the lower bound imposed by the entropy ($H(x)$) [5], as follows:

$$H(x) \leq L < H(x) + 1. \quad (1)$$

Currently, the Huffman's algorithm is widely used to compress data. The algorithm compresses the data by assigning shorter code-words to the most frequent data, whereas the remaining data have longer code-words [2], [3]. Due to the variable length of the code-words generated, this algorithm has high computational requirements. The challenge of this problem is to design and implement a solution that combines the compression performance using Huffman algorithm and an implementation into a small and suitable footprint layout in the multi-channel SAMPA chip. In this study we present an approach to solve these two issues.

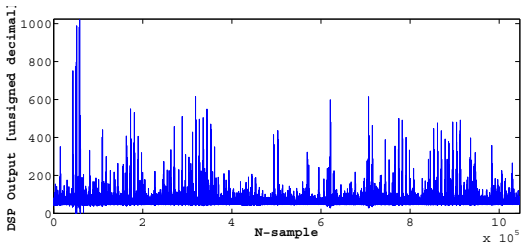
B. Statistical data analysis

The construction of a compression algorithm implies the understanding of the data and its relationship with the precedent hardware; i.e. number of bits at the input, data transmission rate, DC coupling and other characteristics.

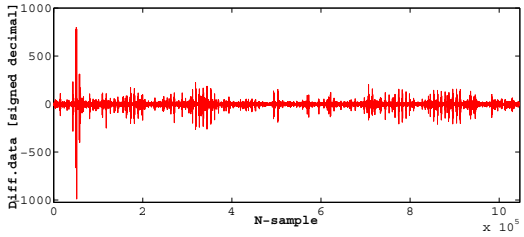
The data in Fig. 3a have 110 miliseconds of real particle collision data, 1099 packets with around 1000 collision-samples each one. These samples were taken at a collision rate of 10kHz and 10MHz of sampling frequency. Considering the upgrade of the experiment, the future packets will be slightly different according to the increase of the collision rate to 50 kHz. However, the sampling frequency will be the same. The data has been shaped by 5495 packets with around 200 samples each one.

Since data is slowly changing and baselines are different for each channel and chip, we applied differential encoding. First,previous incoming data is subtracted from the current-one. With the differential encoding, we were able to get a data stream that is independent from the baseline and therefore we enable the reuse of the Huffman encoder for more channels. Fig. 3b shows the result for DC balancing.

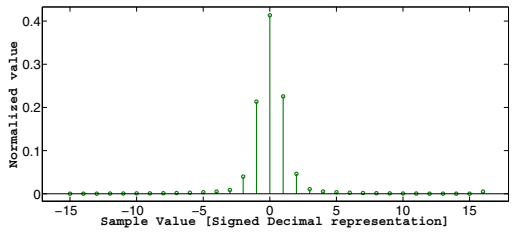
To apply Huffman coding, a probabilistic study of raw data is necessary. The probabilistic study has been done with the data variation obtained in the differential encoding output. The results are shown in Fig. 3c. Note that, approximately 99% of the information is within the range -15 to +15. Therefore, the Huffman coding strategy only encodes the values within the mentioned range and when the difference between the two samples is outside, the compressor assigns a special header, other than Huffman coding, in front of the raw data. This header notifies the receiver that the following 11-bit will be raw data.



(a) DSP raw data output.



(b) Differential encoding for data input (previous incoming data is subtracted from the current one).



(c) Probability distribution of data collisions.

Fig. 3 TPC data plots & probability distribution.

Using the results of the probabilistic study we created a Huffman tree based on one fact: increasing the length of the longest code increases the compression ratio. A Huffman tree coded in MATLAB gives a compression factor of 4.15, but the longest code has 13-bit. There is a tradeoff between the best compression ratio and the maximum length of the longest code.

In this case using a code with 13-bit implies more hardware, rising power consumption and area. So the best choice is to reduce the longest code, then, reducing the compression ratio as well. Final proposed solution have a Huffman tree with compression ratio of 3.89 and the longest code becomes 9-bit long. In Table I shows a portion of the Huffman dictionary. Before sending the data stream, the Huffman code is compacted into a buffer memory of 10-bit wide.

C. Hardware implementation

Fig. 4 shows the general scheme of the compressor. The first part (Register 1, Register 2 and Adder) generates the differential encoding that reduces the DC signal present in the data and allows us to use a unique Huffman dictionary for all collected data. The second part (Encoder) uses a static Huffman dictionary that encodes the differential data generated by the first part. The static dictionary allows us to reduce the number of calculations to encode the data. Finally the

ISBN 978-1-4673-7835-2/16/$31.00©2016 IEEE

IEEE Catalog Number CFP16LAS-ART

TABLE I
HUFFMAN COMPRESSOR DICTIONARY

Data	Probability	Huffman code	Code-length
0	0.41352	0	1
1	0.22571	100	3
-1	0.21323	101	3
2	0.04606	1100	4
-2	0.03987	1101	4
Out of range	0.005099	111000	6

Datapath and FSM (Finite State Machine) allows us to collect compressed data in a 10-bit buffer to send the data when the buffer is full. The FSM controls the flow of data to decide when data is compressed or not, besides verifying the state of the buffer.

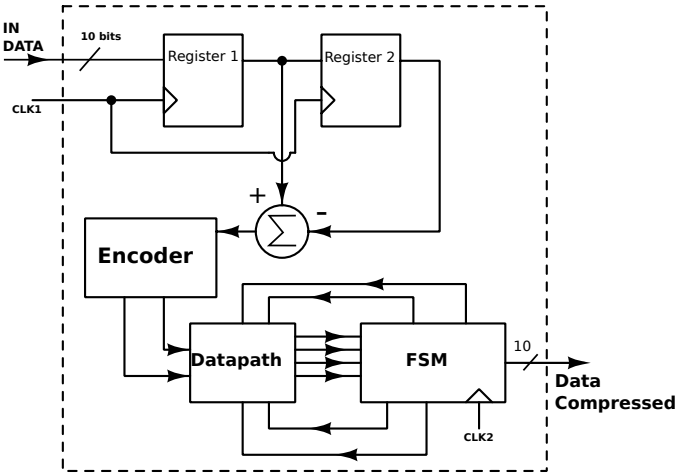


Fig. 4 General scheme of the Huffman compressor implemented on chip.

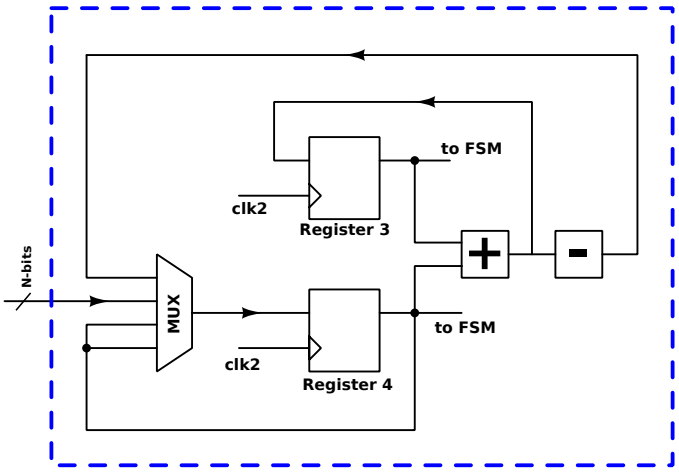


Fig. 5 Basic scheme of datapath embedded on Huffman compressor.

IV. RESULTS AND CHIP IMPLEMENTATIONS

The proposed architecture is fully-synthesized on 130nm and 65nm CMOS technology. The cells report shows that both

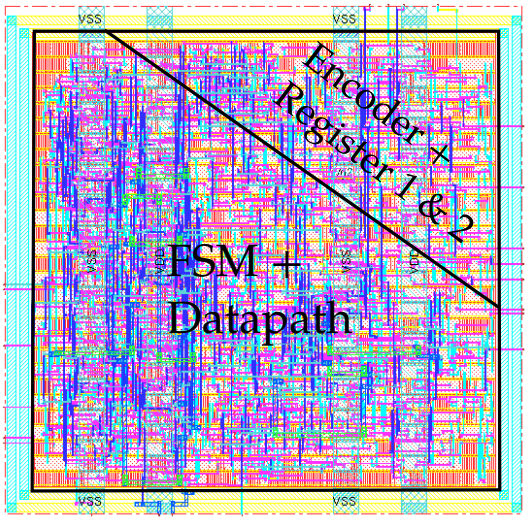


Fig. 6 Layout photograph using 65nm CMOS technology, 60μm×60 μm.

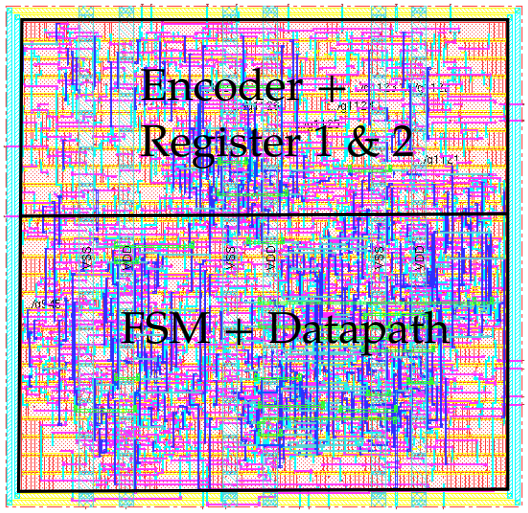


Fig. 7 Layout photograph using 130nm CMOS technology, 105μm×105 μm.

TABLE II
CHIP SPECIFICATIONS

Parameters	IC 1*	IC 2*	IC 3 [4]
Process [nm]	65	130	90
Layout Area [μm^2]	60x60	105x105	78.9x75.6
Power [mW]	1.2	1.9	0.2
Max. Frequency[MHz]	435	333	-
Clock Rate[MHz]	160	160	200

*Measures taken at TT typical corner process, 1.2[V] and 25°C

technologies used 62 sequential gates. However, the 65nm technology used more logic gates than 130nm technology (663 against 651).

Although we use different library corners for synthesis, we report results of the typical case library with the typical process, a 1.2V supply voltage and a temperature of 25°C for

ISBN 978-1-4673-7835-2/16/$31.00©2016 IEEE

IEEE Catalog Number CFP16LAS-ART

both 65nm and 130nm technologies. Table II summarizes the performance of the architecture showing that the maximum operation frequency for 65nm is 435MHz and for 130nm is 333MHz whereas the power consumption is 1.2mW and 1.9mW respectively. The Fig. 6 and Fig. 7 show the final layout for both 65nm and 130nm technologies with dimensions of 60μm$\times$60μm and 105μm$\times$105μm respectively.

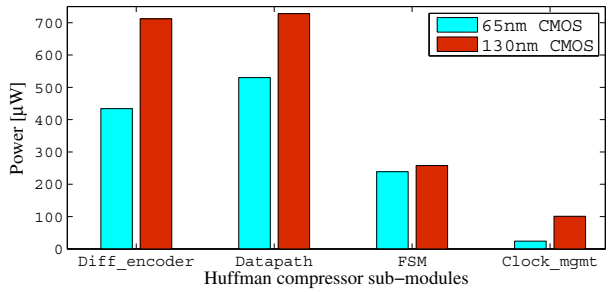


Fig. 8 Power breakout of Huffman compressor.

Fig. 8 shows the power consumption of each sub-module into the Huffman Compressor. The datapath sub-module consumes the biggest amount of power in both technologies because this module has many registers involved in the load of data in the output buffer. Another module with high power consumption is the Diff_Encoder (Differential Encoding & Encoder) which saves the Huffman dictionary in registers to compare and compress data.

Reported work of Huffman compressor have been implemented for different applications. In [4] the idea is to compress data from an electrocardiograph (ECG) in wearable devices. This encoder is focused on pre-processed compressed ECG signals which are on the SLF (Super Low Frequency). On the other hand, the proposed design in our study compresses signals from a particle collider which are on the LF (Low Frequency) band. The proposed encoder uses a high frequency clock due to the need of processing the data before encoding it, hence it is not suitable to compare both designs based on frequency or power consumption. However, the proposed design consumes more power than the ECG compressor due to the usage of a lot of registers for output data purposes which can be reduced using fewer number of registers for the dictionary.

A comparison of the proposed encoder with the ECG compressor can be made through the proposed Figure of Merit (FoM) in (2) including the area of final layout. Table III shows a general comparison between the three Integrated Circuits using the FoM. The ECG compressor has a better FoM due to the low power consumption but it is important to highlight that the proposed encoder includes part of the communication process.

$$Figure\ of\ merit = \frac{Frequency}{Area * Power} \qquad (2)$$

TABLE III
PERFORMANCE COMPARISON.

Process [nm]	65	130	90 [4]
Figure of Merit	37.04	7.64	166.81
Figure of Merit (Normalized)	0.22	0.05	1

V. SUMMARY

The trade off between code-length and compression ratio is highly important to consider when a Huffman algorithm is used. Reducing the longest code length allows to save area but this reduction decreases the compression ratio. A Huffman enconding scheme with small footprint has been proposed with a compression ratio that enables the increase of data rate density to about four folds in current electronics at the LHC. Area and power results of the proposed scheme show that it is possible to implement a compression scheme inside the readout chips of the ALICE project at CERN since no data is lost and therefore full-data is later processed in data centers.

REFERENCES

[1] The ALICE Collaboration. "Technical Design Report for Upgrade of the ALICE Read-out & Trigger System". July 2014.[online] https://cds.cern.ch/record/1622286/files/ALICE-TDR-016.pdf , pp 63-68
[2] D.A. Huffman, "A method for the construction of minimum-redundancy codes", *Proceedings of the I.R.E.*, Sept 1952, pp. 1098-1102.
[3] David. Salomon, *Data Compression: The Complete Reference*. New York: Springer-Verlag (4th ed.), 2007, pp. 74-95.
[4] Shih-Lun Chen, Min-Chun Tuan, Tsun-Kuang Chi and Tin-Lan Lin,"VLSI architecture of lossless ECG compression design based on fuzzy decision and optimisation method for wearable devices", *Electronics Letters*, vol. 51, no. 18, pp.1409-1411, 2015.
[5] Cover, Thomas M and Thomas, Joy A., *Elements of information theory*, New Jersey: John Wiley & Sons, 2012, pp. 112-113, 123-127.

ISBN 978-1-4673-7835-2/16/$31.00©2016 IEEE

IEEE Catalog Number CFP16LAS-ART