JPL Publication 91-13

A Very High Speed Lossless Compression/Decompression Chip Set

Jack Venbrux
Norley Liu
Kathy Liu
Peter Vincent
Randy Merrell

NASA Space Engineering Research Center
for VLSI System Design, University of Idaho

July 15, 1991

NASA
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 91-13

A Very High Speed Lossless Compression/Decompression Chip Set

Jack Venbrux
Norley Liu
Kathy Liu
Peter Vincent
Randy Merrell

NASA Space Engineering Research Center
for VLSI System Design, University of Idaho

July 15, 1991

NASA
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

The research described in this publication was carried out by the NASA Space Engineering Research Center for VLSI System Design, University of Idaho, under a contract with the National Aeronautics and Space Administration, together with Goddard Space Flight Center. Additional technical suggestions were provided by the Jet Propulsion Laboratory, California Institute of Technology.

ABSTRACT

This publication describes a chip set that implements the Rice Algorithm for lossless compression and decompression of source data. The Rice Algorithm performs adaptive lossless compression and decompression yielding an efficient performance over a wide entropy range. The algorithm has two particularly useful features from an implementation point of view. It requires no table lookups even though its performance spans a wide entropy range. Furthermore, because many of the coding functions are similar in nature, a significant amount of internal hardware was shared in the compressor. Likewise, the decompressor has shared hardware due to similarities between the decoding functions. All of the hardware sharing that was done while putting the Rice Algorithm into silicon resulted in small chips, both being only 5mm on a side.

The chip set has been designed in a 1 micron CMOS process for low power consumption and high data rates. The expected power dissipation for each chip is less than 1/4 Watt while operating at maximum clock rate. The compressor has a designed data rate of 20 megasamples/second at military specifications. At maximum quantization (n=14 bits/sample), this corresponds to 280 Mbits/second or approximately 500 Mbits/second under nominal conditions. The decompressor, which is a more complicated design, can decompress data at 10 megasamples/second at industrial specifications.

At present, the compressor chip requires a packetizer at its output. The packetizer concatenates blocks of compressed data into whatever size packets are required by system designers. This allows for maximum flexibility in packet construction, but requires a packetizer and an unpacker–both of which are in the process of being designed at Goddard Space Flight Center. The decompressor is designed so that it will decompress either packetized data or data that is not in packet form. Using packetized data over non-packetized data has an important advantage: the decompressor will not allow any errors in transmission to propagate between packets.

The chip set has been designed to be applicable to a wide variety of compression needs. The data format used by the chip set is being considered as a possible Space Data Systems standard for source coding. Both chips support a quantization range from 4...14 bits/sample and can use two different types of prediction. The default prediction method built into the chips is nearest neighbor prediction, but external prediction is also supported. This allows the chip set to be used with prediction in the Y and Z dimensions.

An additional feature extends the generality of the chip set by allowing the predictive pre-processing part to be bypassed. A user can model any type of data, and then feed the non-negative results directly into a separate adaptive variable length coding section. The chip set is then used as a high speed entropy coder/decoder without being tied to a particular modeling technique. Entropy coders, for example, are often used as the last stage in a lossy compression system to boost performance.

ACKNOWLEDGEMENT

The authors wish to thank Warner Miller and Pen-Shu Yeh at Goddard Space Flight Center for their support, encouragement, and for providing direction throughout the course of this project. Without them, this project would not have been possible. The authors also wish to thank Robert F. Rice at the Jet Propulsion Laboratory for his encouragement and helpful suggestions throughout the project, and for his help in making this a more readable publication.

Contents

List of Figures

List of Tables

A Very High Speed Lossless Compression/Decompression Chip Set

1 Introduction

The chip set discussed in this publication[1] is in fabrication and is designed to perform high speed lossless compression and decompression. The algorithm used is the Rice Algorithm [1,2], which adapts rapidly to changing entropy conditions and efficiently compresses data over a wide entropy range. The chip set implements some of the algorithms described in the publication, "Algorithms for a Very High Speed Universal Noiseless Coding Module [3]." Recent work at the Jet Propulsion Laboratory (JPL) has produced working Rice encoders [4]. This work differs from the JPL project in that it is a more general design, includes a decoder, and is designed to operate at a higher speed. The data format used by this chip set is also being considered as a possible Space Data Systems standard for Source Coding [5].

One of the objectives for designing and fabricating the chip set is to provide real time image compression for Earth-orbiting satellites. The chip set is designed to handle quantization from 4 through 14 bits and allow for external prediction as well as nearest neighbor prediction. The Encoder is designed to operate at 20 megasamples (Msamples) per second at military specifications. At maximum quantization, that corresponds to 280 Mbits of lossless compression per second. *Under nominal conditions, the rate should be approximately 500 Mbits/second.* The decoder, which is a more complex design, can decompress data at a data rate of 10 Msamples/second at industrial specifications. Both chips are being fabricated in a 1 micron CMOS process.

2 Overview of Rice Algorithm

The Rice Algorithm is a lossless compression method that is efficient over a wide range of entropy conditions. The algorithm handles different entropy conditions by utilizing multiple coders, each of which is tuned to compress data at a particular entropy range. Instead of *guessing* which of the coders might do the best job on the next block of samples, as do many compression techniques, this algorithm codes a block of samples with all of the coders simultaneously, and then selects the output from the coder that does the best compression job. An interesting feature of the algorithm is that it compresses data over a wide entropy range *without using any lookup tables.*

Compression is done on small blocks of samples (16, for example). This provides excellent adaptation. Coding samples into large blocks, such as an image scanline, would force the coders to pick the winner based on the average entropy rather than tracking the entropy changes within the scanline. Smaller blocks do have slightly increased overhead due to the ID bits (identification bits) preceding each data block. The ID bits inform the

[1]This research was supported in part by NASA under grant NAGW-1406.

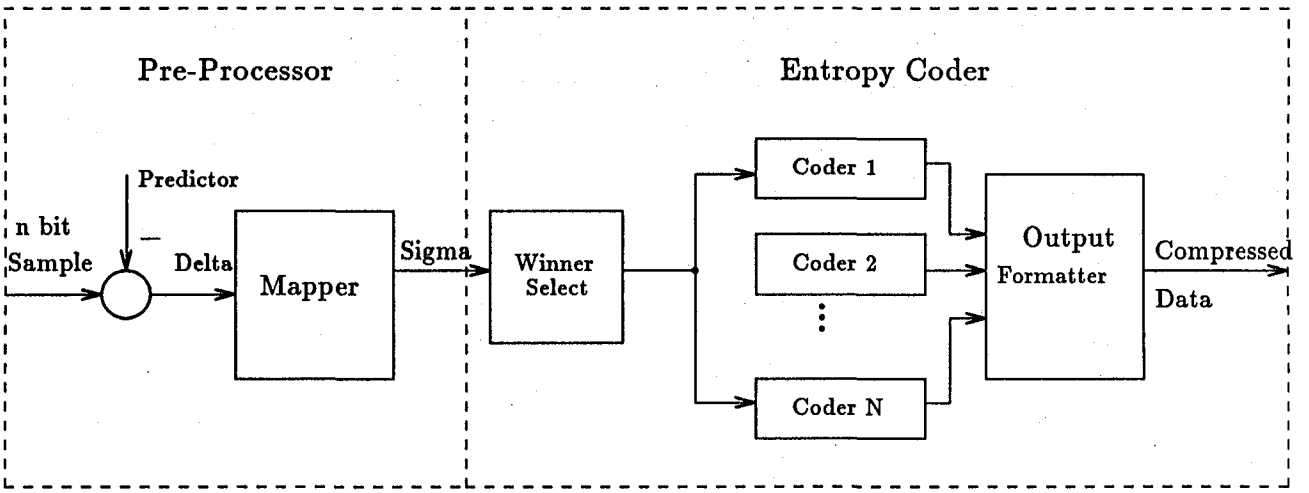


Figure 1: Block Diagram of Rice Algorithm Architecture

decoder which of the coders was selected for that block. The decoder must then decode the block based on that particular coder's algorithm.

A block diagram of the Rice Algorithm is shown in Figure 1. The major blocks consist of a pre-processor followed by an adaptive entropy coder. To explain the operation of the pre-processor, assume that the Rice Algorithm coder is being used in a typical application. In a typical application, the first step in the encoding process is to save a reference sample at the start of every long data block such as a scanline. This reference will be used to predict the next sample. The predictive coder takes the difference, labeled the Delta value, between the present sample and the predictor. When using nearest neighbor prediction, the previous sample is the predictor, however, the chip set also supports using an externally supplied predictor for prediction in the Y or Z dimensions.

Subtracting two n bit values, the present sample and the predictor, results in a difference of n+1 bits. The mapper's function is to reduce the difference back down to n bits before coding. The pre-processor, with the predictive coder and mapper, supplies the entropy coder with numbers of the following form: non-negative integers where the smaller integers are more likely than the larger.

The entropy coder begins with a winner select operation. It may seem odd to choose the winner before coding, but that is one of the advantages of the algorithm. By simple shifts, additions, and comparisons, the winner is calculated *before* any coding is done. This allows for sharing hardware among the many parallel coders. One particular coder, the default coder, limits expansion under unexpected entropy conditions. If the encoder cannot compress the data, the default coder sends out ID bits followed by uncompressed data. Many other coding methods cause significant data expansion when the entropy is too high.

The encoder's pre-processor, with the predictive coder and mapper, can be bypassed. This allows a user to model any type of data, even if it doesn't fit into a predictive coding scheme, and then feed non-negative numbers directly into the entropy coder.

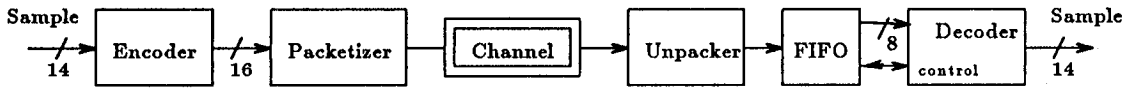


Figure 2: System Diagram

Likewise, the decoder's post processor can be bypassed (see Figure 7). *When operating in the pre-processor-bypass-mode the chip set is acting as a fast general purpose entropy coder/decoder.*

3 The System

There are five major components needed for the Data Compression system:

1. Data Compression Encoder
2. Packetizer
3. Unpacker
4. FIFO
5. Decoder

The basic system is shown in Figure 2. After the encoder compresses a block of 16 samples, it outputs the block as a sequence of 16 bit words. A packetizer receives the data and concatenates many blocks together to form a packet. The packetizer adds a header before the packet is transmitted through a channel to an unpacker. The unpacker strips away header information from the packet leaving only encoded sample data, which is then written to a FIFO.

The decoder reads and processes the data stored in the FIFO at a rate at least equal to the encoder's coding rate. The decoder can decode data from a variety of packet types including fixed and variable length packets. Should the system truncate a packet, as might happen when the compression ratio is not high enough, the decoder will still recover as many samples as possible. The decoder design also handles data that is not in packets.

Like most compression schemes, the decoder will get lost in the presence of errors. Error correction coding should be used in the majority of applications to remove errors of transmission from the data to be processed. *One of the features designed into the decoder, however, is that errors will not propagate between packets.*

4 Compression Performance

The encoder and decoder implement a Rice coding set that has 12 different coders. Those 12 coders allow for efficient compression over an entropy range of approximately 1.5...12.5 bits per sample. A theoretical performance graph of the chip set is shown in Figure 3. The straight line is the theoretical entropy calculated using a first order Markov model. The actual performance is close to the theoretical on all but the very lowest and

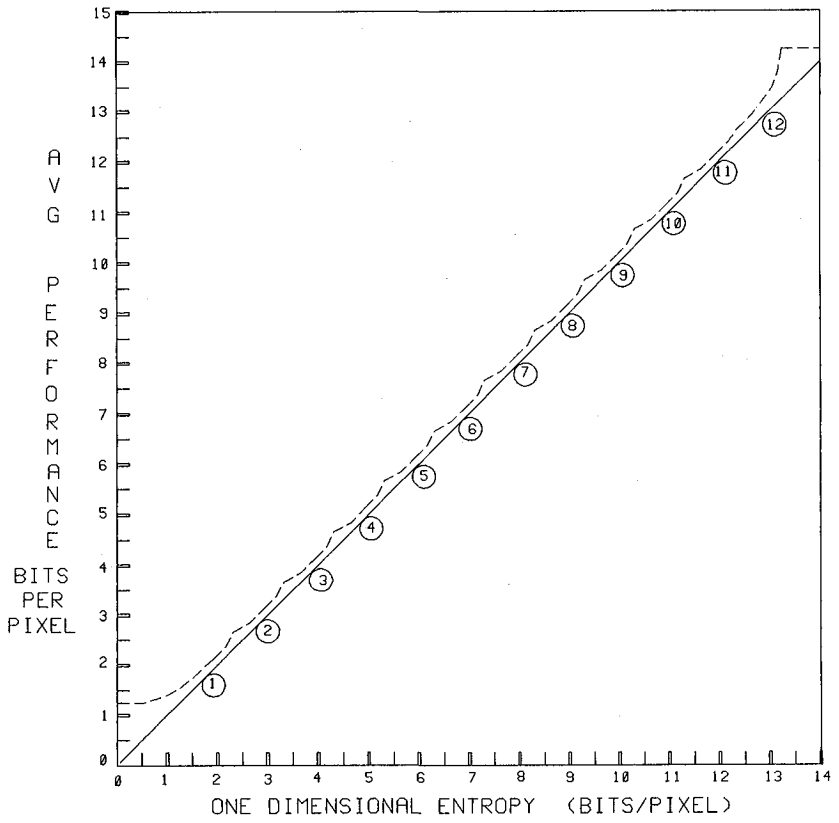


Figure 3: Chip Set Performance versus Entropy with 12 Rice Coders.

highest entropy conditions. The bottom of the performance curve is one quarter of a bit per sample above the theoretical due to the ID bits that must precede each block of data. The graph is based on work done at Goddard Space Flight Center which showed that these code options are equivalent to Huffman codes [6]. The graph includes 12 circled numbers. The placement of the numbers corresponds to the center of the entropy range covered by that particular coder.

The amount of compression achieved depends on the entropy of the source being compressed. The 12 option coding set has been simulated using actual satellite images from different instruments. At 12 bits quantization, the encoder is expected to achieve a compression ratio of approximately 2 to 1 when compressing most earth images [6].

5 Data Rates

The encoder is designed to compress data at 20 Msamples/second at military specifications (125C, 4.5 Volt supply, and 3 sigma worst case processing). For 8,10,12, and 14 bit quantization the corresponding data rates in Mbits/second are shown in Table 1. At room temperature, with a 5 Volt supply, and typical processing, the data rates are expected to be approximately 1.8 times those listed under military specifications. The decoder's designed decoding rate is 10 Msamples/second at industrial specifications (75C, 4.75 Volt

Bits per	*data rate*	*Mbits/sec*
Sample	*MIL spec*	*Nominal*
8	160	288
10	200	360
12	240	432
14	280	504

Table 1: Data Rate Estimates at Different Quantization Levels.

supply, and 3 sigma worst case processing). A system could be made to operate at 20 Msamples/second but would require paralleling 2 decoders and adding some extra logic and memory.

6 The Chip Set

The encoder and decoder chip set is being fabricated in a 1.0 micron CMOS process. The chip layouts are shown in Figure 4. Careful attention was paid to minimizing noise on the power grids and incorporating layout techniques that reduce the chance of latch-up. Due to performance requirements, a custom design approach was used to implement the chips. Both chips have less than 37,000 transistors and are only 5mm on a side.

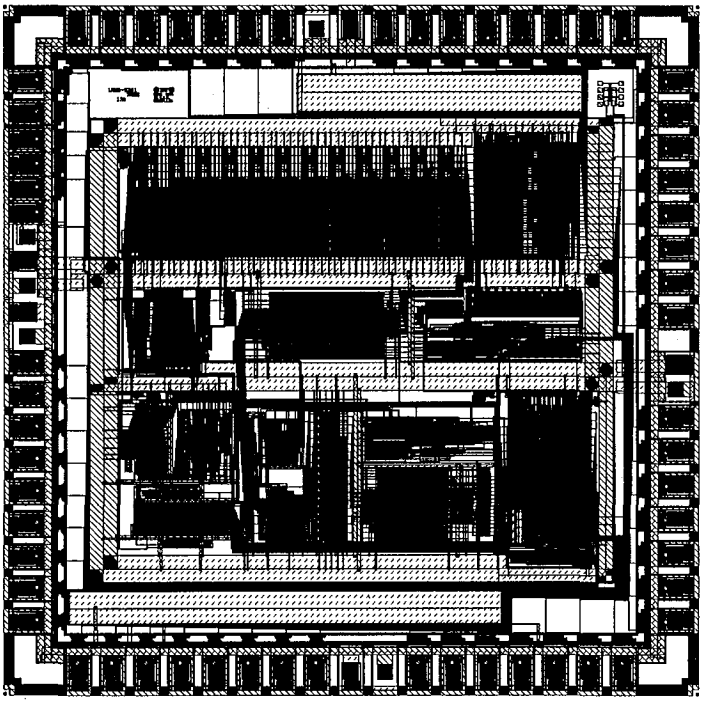
The chip set is not designed to withstand SEUs (single event upsets), but testing has shown that the process is able to survive 1 megarad total dosage. To allow for testing radiation effects on these particular chips, six transistors were added to each design. Small, typical, and square (20uM x 20uM) N and P devices were included with node access by probe pads. To limit the chance of ionic contamination within the core of the chip, these pads were covered by passivation. Access to them would be by persistent use of a probe or by etching off the passivation layer.

One feature of this chip set is that reference samples are inserted automatically into the encoder's output data stream. The alternative, letting the external system handle references, places a burden on system design that extends from the encoder to the decoder. With this chip set, a user sets how often to insert reference samples.

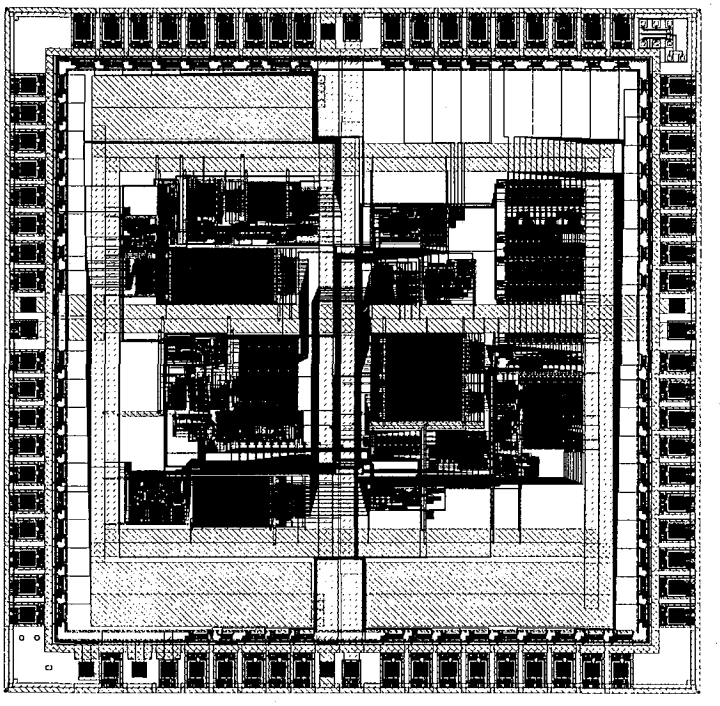
6.1 Encoder

The encoder architecture is a true pipeline with data being input every clock cycle and compressed data being output when it is available. It takes 63 clock cycles to fill up the encoder, on the 64th clock cycle data is output. The 64 clock cycle latency is independent of the winning coding option and was fixed to simplify system design.

The major sections, shown in Figure 5, are similar to the Rice Architecture shown in Figure 1. The major difference between the two figures, is the many parallel coders shown in the Rice Architecture diagram have been replaced with just two coding blocks.



(a) Encoder Layout.



(b) Decoder Layout.

Figure 4: Chip Set Layout.

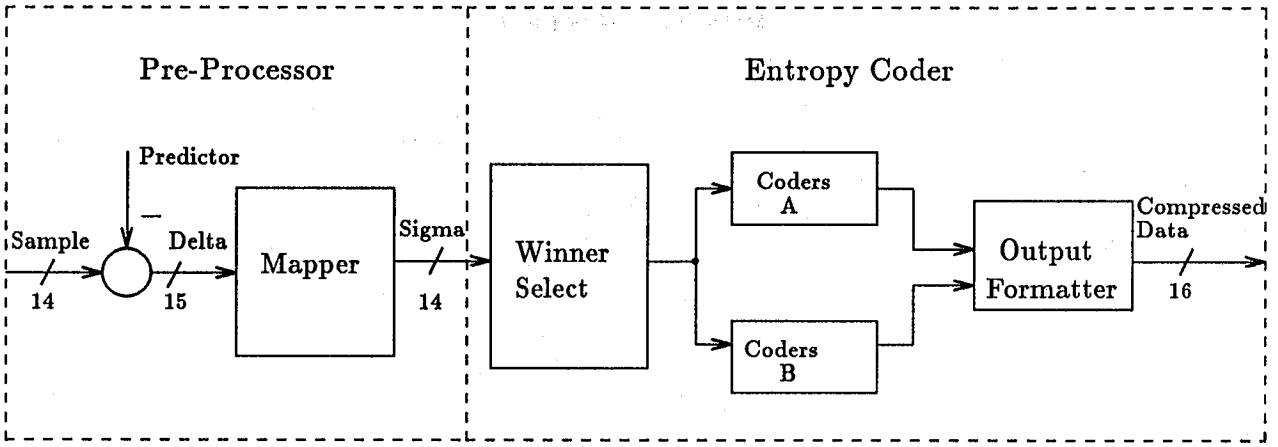


Figure 5: Block Diagram of Encoder Architecture.

Grouping 12 different coding options into 2 coding blocks was possible because 11 of the 12 coders perform very similar operations, which allowed sharing much of the hardware.

The encoder chip layout plan is shown in Figure 6. The MAPPER block performs the differencing and mapping of incoming samples. CNTBLK, the largest single block in

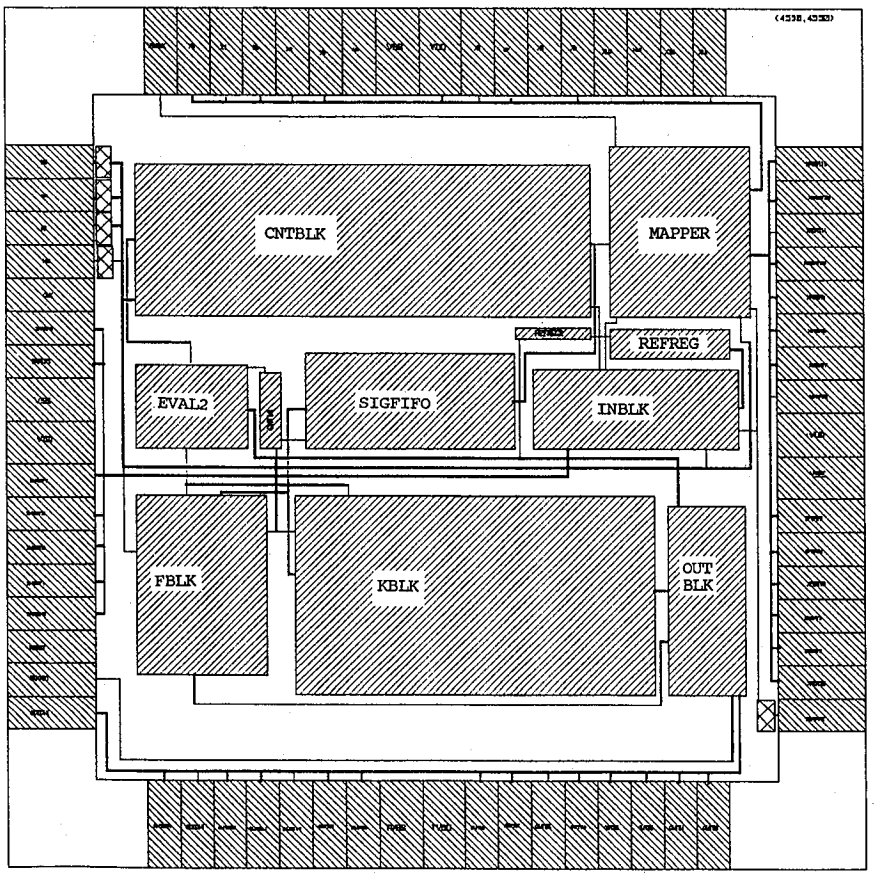


Figure 6: Encoder Layout Plan.

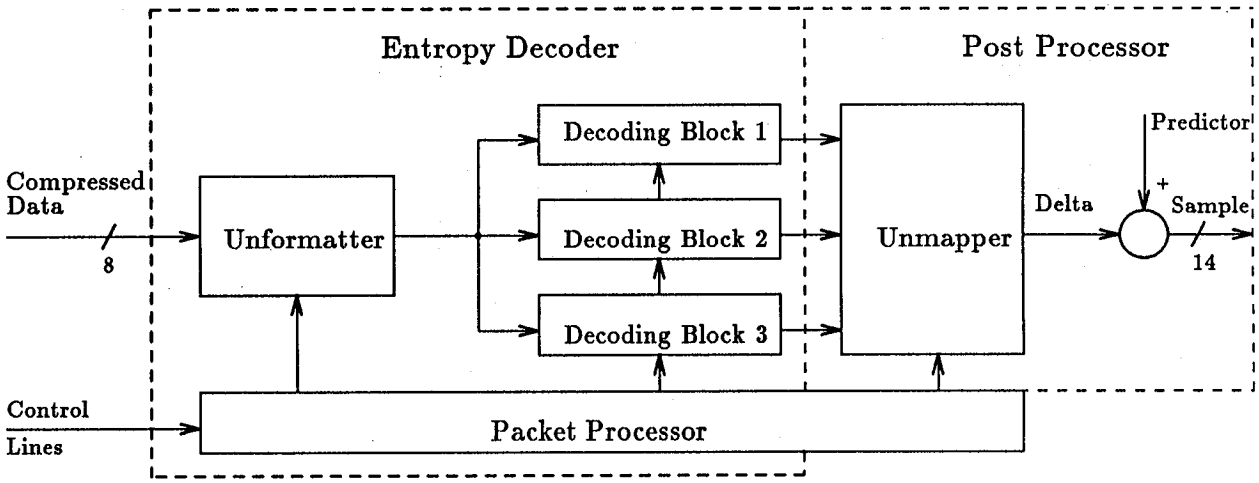


Figure 7: Block Diagram of Decoder Architecture.

the chip, selects the winning option. It uses exact counts to select the winner instead of using approximations [3]. The two coding blocks, FBLK and KBLK, perform 12 different coding operations. Output formatting is done primarily in OUTBLK. INBLK is a control section, SIGFIFO stores sigma values until the winner is known, and EVAL2 performs some calculations needed by the coding blocks. Most of the control is distributed along the data paths with some random logic in INBLK, FBLK, and KBLK sections. The random logic throughout both chips was reduced by using state machines that use a special binary tree structure developed at the University of Idaho [7].

6.2 Decoder

Because the Rice Algorithm is a serial algorithm, decoding at a high data rate is more difficult than encoding. The major bottleneck in the decoding process is that the position of a new symbol in the data stream cannot be decided until the previous sample is decoded. A block diagram of the decoder is shown in Figure 7. The process of decoding is the reverse of encoding. First, the data must be unformatted by reading which winning ID is at the front of each block. Then, depending on the ID, the data is decoded in one of 12 ways. Like the encoder, the decoder shared hardware so that only 3 data path blocks were needed to perform the decoding instead of 12. After decoding, the data is unmapped, formatted and output.

The layout plan for the decoder is shown in Figure 8. The IFORM and WDSEL sections comprise the Unformatter shown in the block diagram. The three decoding blocks were broken up and pipelined into the two blocks labeled DECOD1 and DECOD2. The unmapping function and summing the predictor with the unmapped value are functions performed in the layout block OFORM. Finally, the TIMING section provides global control for various sections of the decoder's pipeline.

Two very specialized serial to parallel converters were designed and used in the unfor-

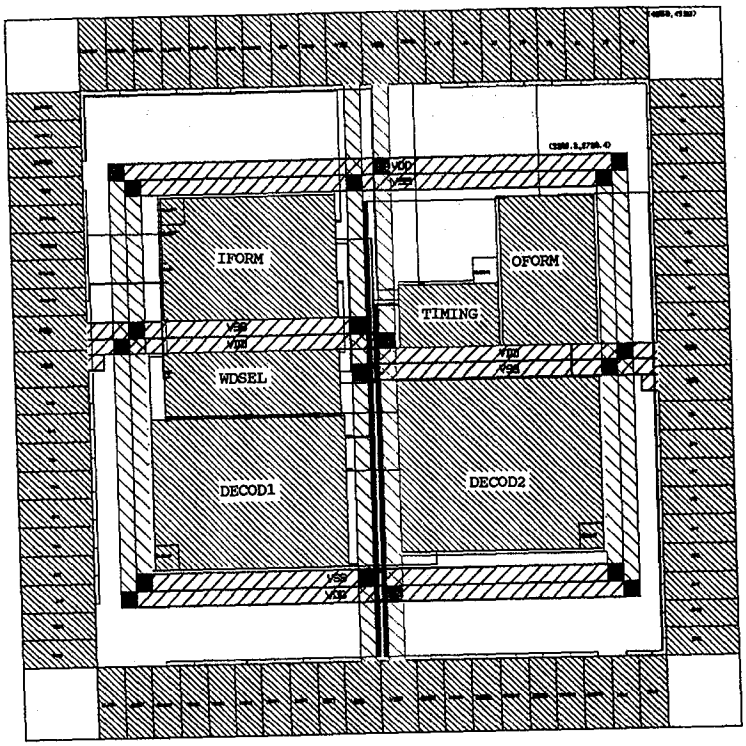


Figure 8: Decoder Layout Plan.

matter and decoding blocks to meet speed requirements. The special shifters take up a large part of the area in those sections. Some of the control complexity of the decoder is due to handling the many options, including different types of packet modes, continuous mode, varying quantization values, and different predictors. The decoder recovers all samples not affected by the truncation, and then signals that truncation has occurred while it continues decoding. The decoder will output dummy pixels until the packet is filled out. Handling packet truncation accounts for a large part of the complexity of some control sections.

7 Comparison with a Microprocessor

In a report written for Goddard Space Flight Center, one of the authors of this publication compared implementing the Rice Algorithm using an 80386 microprocessor with using the custom Rice encoder [8]. That comparison was based on analyzing the number of instructions required to perform the typical coding options of the Rice Algorithm for 12 bit samples. It was assumed that all code could be written in line, that there were no wait states of any kind, and all operations could be performed in 2 clock cycles (the speed of register operations). With this optimistic scenario, it would take 740 instructions to compress one block of 16 samples. With each instruction taking 2 clock cycles, a 16 Mhz 80386 microprocessor could compress data at 2.08 Mbits/second. In actual practice, the 80386 would not be able to compress data at this rate primarily due to the occasional need to access memory, which takes twice as long as register operations.

Comparison	*80386*	*ENC*	*80386 vs ENC*
speed (Mbits/sec)	2.08	240	1:115
power (unloaded)	1.7W	0.17W	10x
transistors	375,000	37,000	10x
max internal clock	16 Mhz	20 Mhz	-

Table 2: Comparison of Maximum throughput of an 80386 and the Rice Encoder for 12 bits per sample data.

As Table 2 indicates, *even though the Rice Encoder (ENC) can compress data 115 times faster than the microprocessor it still only uses 1/10th the power.* The power comparison figure does not include any of the support chips needed by the microprocessor such as RAM and ROM. Adding RAM and ROM would increase the power consumption of the microprocessor implementation to a value much higher than the 1.7 Watts shown in Table 2. The microprocessor also has 10 times more transistors than the encoder chip. Fewer transistors and reduced chip complexity permit the creation of test patterns that toggle over 94% of the encoder's nodes– something that is almost impossible to do with a typical microprocessor.

Table 3 shows a comparison of the 80386 microprocessor processing data at its estimated maximum rate of 2.08 Mbits/second, with the custom encoder running at the same bit rate. In order to slow the custom encoder chip down to 2.08 Mbits/second it must operate at 173Khz– a clock frequency 92 times slower than the processor. Because of the slower clock and smaller size, *the Rice Encoder's unloaded power consumption is 850 times less than the general purpose microprocessor.*

Comparison	*80386*	*ENC*	*80386 vs ENC*
Mbits/sec	2.08	2.08	-
clock	16 Mhz	173 Khz	92x clk freq.
power (unloaded)	1.7W	0.002W	850x power

Table 3: Comparing the Intel 80386 processor with the Rice Encoder compressing at the same bit rate.

8 Summary

A chip set has been described that will perform lossless compression and decompression using the Rice Algorithm. The chip set is designed to compress and decompress source data in real time for many applications. The encoder is designed to code at 20 Msamples/second at MIL specifications. That corresponds to 280 Mbits/second at maximum

quantization or approximately 500 Mbits/second under nominal conditions. The decoder is designed to decode at 10 Msamples/second at industrial specifications. A wide range of quantization levels is allowed (4...14 bits) and both nearest neighbor prediction and external prediction are supported. When the pre and post processors are bypassed, the chip set performs high speed entropy coding and decoding. This frees the chip set from being tied to one modeling technique or specific application.

Both the encoder and decoder are being fabricated in a 1.0 micron CMOS process that has been tested to survive 1 megarad of total radiation dosage. The CMOS chips are small, only 5mm on a side, and both are estimated to consume less than 1/4 of a Watt of power while operating at maximum frequency.

[1] R. F. Rice, "Some Practical Universal Noiseless Coding Techniques", JPL Publication 79-22, Jet Propulsion Laboratory, Pasadena, California, March, 1979.

[2] R. F. Rice and Jun-Ji Lee, "Some Practical Universal Noiseless Coding Techniques, Part II", JPL Publication 83-17, Jet Propulsion Laboratory, Pasadena, California, March 1983.

[3] R. F. Rice, Pen-Shu Yeh, and W. H. Miller, "Algorithms for a Very High Speed Universal Noiseless Coding Module", JPL Publication 91-1, Jet Propulsion Laboratory, Pasadena, California, February 15, 1991.

[4] R. Anderson, et al., "A Very High Speed Noiseless Data Compression Chip for Space Imaging Applications", JPL Publication 91-12, Jet Propulsion Laboratory, Pasadena, California (to be published).

[5] W. H. Miller, et al., "CCSDS Recommendation for Universal Source Coding for Data Compression", White Book, National Aeronautics and Space Administration, Washington, D.C., May 1991, Preliminary.

[6] Pen-Shu Yeh, R. F. Rice, and W. H. Miller, "On the Optimality of Code Options for a Universal Noiseless Coder", JPL Publication 91-2, Jet Propulsion Laboratory, Pasadena, California, February 15, 1991.

[7] S. Whitaker and G. Maki, "Sequence Invariant State Machines", NASA Symposium on VLSI Design, January 1990, pp. 241-252.

[8] J. Venbrux, "Implementing the Rice Algorithm Using an 80386 Microprocessor as Compared to Using a Custom IC", Contract report submitted to Goddard Space Flight Center under NASA grant NAG5-1043 (internal document), Goddard Space Flight Center, Greenbelt, Maryland, November 21, 1990.