An ASIP design for low loss compression of front-haul data in 5G base stations

Yunxiang Tang, Biao Long, and Dake Liu

Abstract  With the continuous development and commercialization of 4G and 5G technology, the bandwidth demand of front-haul IQ data in CPRI/eCPRI increases rapidly, so data in CPRI/eCPRI links need to be compressed. To solve this problem, this paper built a compression/decompression ASIP (Application Specific Instruction-set Processor) module as a subsystem of our front-haul processor system which could choose algorithms flexibly and data compression ratio with low delay and low distortion. Its advantage is that the three compression algorithms are integrated together, and the compression precision is improved through the study of distortion minimization. Through the performance evaluation, the data transmission rate can reach 537.6 Gbps, and the hardware overhead and EVM value are both satisfied.

Keywords: CPRI/eCPRI, data compression, μ-law, block floating point, ASIP module

Classification: Microwave and millimeter-wave devices, circuits, and modules

1. Introduction

Starting from 3G, mobile communication operators and infrastructure suppliers divide wireless base stations into two parts: Baseband Unit (BBU) and Remote Radio Unit (RRU) [1], and define a standard interface between them: Common Public Radio Interface / evolved Common Public Radio Interface (CPRI / eCPRI) [2, 3]. With the commercial use of 5G technology with massive MIMO technology [4] as the core, the bandwidth of CPRI / eCPRI standard has nearly doubled every two years, and the required rate has reached 25Gbps in 2020. So the bandwidth resource problem becomes obvious [5]. It is important to design quality data compression modules for base station front-haul, to keep low EVM and price of interface hardware at the same time. In order to reduce the rate requirements between BBU and RRU, in addition to adopting more efficient coding methods, data compression is also required [6]. There are two compression types and two scenes: lossless compression and lossy compression, time domain (Option-8) and frequency domain (Option-7) [7].

The work of this paper was to achieve a low-cost, low-delay, relatively general front-haul data compression and decompression module applied to base station. The compression scheme was to add compression and decompression processing modules to BBU and RRU respectively, as shown in Fig. 1 [8]. This paper used three compression algorithms, single signal compression based on μ-law, IQ signal compression based on μ-law and block floating point (BFP) compression. Then we integrated the three compression algorithms, defined the instruction set, and designed an instruction set architecture as a 5-layer pipeline architecture. Multi-modes ASIP architecture supporting multiple algorithms was proposed. By using different assembly instructions, different compression rates, compression methods and saturation methods could be configured, so that users can flexibly choose and configure the best compression method. Compression significantly reduce the CPRI data rate and enable operators to reduce the construction cost of base stations. It meets the requirements of open wireless alliance (ORAN), and takes a big step forward to the goal of front-haul interworking communication of 4G / 5G base stations.

In most of the references, the data compression methods include reducing the number of samples and bits represented per sample [9]. We still have doubts about the K / L downsampling method, because the input signal needs to be fidelity, so it is not suitable to use K / L downsampling in 4G and 5G front-haul, which will reduce the fidelity of the signal.

The focus of this paper is to balance data compression and EVM. In reference [10], the improved IQ data compression technology based on μ-law is also applied and achieved good results. However, it only discusses the compression in the time domain, and does not discuss the compression in the frequency domain, nor does it discuss the influence of different modulation methods on the compression accuracy. In order to reduce EVM, a nonlinear quantization method [11] is proposed based on the assumption that the input signal distribution feature is Gaussian distribution. Compared with the above compression methods, the block scaling combined can meet the above requirements. In references [12, 13, 14, 15, 16, 17, 18], the sample in a block is normalized by the sample with the largest size, which is...
called the scaling factor, and then the normalized sample is quantified.

2. Basic algorithm

In order to meet the transmission requirements and reduce the cost, this paper selected and modified three baseband signal compression methods with low delay and high rate, which were suitable for 4G and 5G.

2.1 μ-Law compression/decompression algorithm

The μ-Law compression algorithm originated from the ITU-T G.711 standard [19], which like A-Law, was a set of speech compression standards developed by the International Telecommunication Union. In this paper, we improved the μ-law algorithm according to the need.

In fact, in the ORAN standard, only the μ-law compression algorithm is used (the μ-law compression formula is shown in the following figure).

\[ f(x) = \frac{\ln(1 + \mu x)}{\ln(1 + \mu)} \quad 0 \leq x \leq 1. \quad (1) \]

where \( x \) is the input signal, \( f(x) \) is the output signal, and \( \mu \) is the compression parameter of μ-law. The value of \( \mu \) is usually 200.

This paper used a dynamic bias compression algorithm, which could automatically select the appropriate bias size according to different input values, so we get high accuracy of compression.

2.2 Block floating point compression/decompression algorithm

This algorithm was based on the requirements of ORAN-WG4.CUS.0-v01.00 standard [20] about block floating point compression. This compression requirement was described in the standard file of O-RAN. The format of the compressed data is shown in Fig. 2 [20]. For each Physical Resource Block (PRB), floating point data of in-phase (I) and orthogonal (Q) samples are transmitted. Then, the samples are compressed into sign bits, mantissas, and a common exponent. Then we used the common exponent to compress these baseband IQ data at the same time. In the block floating point algorithm, we also added a dynamic bias algorithm to improve the accuracy of the algorithm [20].

2.3 IQ compression/decompression algorithm

The IQ compression algorithm was a complex signal compression algorithm [21] and it was similar to the block floating point algorithm. In the IQ compression algorithm, the exponents of the real and imaginary parts of the data were shared, and the μ-Law algorithm was used for each compression, which was a combination of the first two algorithms.

3. Design and implementation of hardware module

3.1 Algorithm implementation

In the algorithm implementation stage, this paper used the modular design concept. The algorithm was divided into several small modules, so that some small modules could be used in a variety of algorithms. Reasonable module division could greatly reduce the algorithm area overhead.

Many modules in the three algorithms were similar, such as rounding, exponent generation and comparison, Mantissa generation, and data recovery. Therefore, we completed the partial reuse of the circuit in these modules.

The pseudocode of the core algorithm is as follows:

3.2 Method of distortion minimization

In order to approach the limit, the method of rounding [22] and bias was added in this paper, which greatly improved the accuracy after compression and obtained satisfied results.

The principle of bias is as follows: Due to the non-linear characteristics of the compression algorithm, the quantization interval of large signals is larger than that of small signals, which leads to low compression accuracy of large signals. According to the different exponent of the input

Algorithm 1 Compress pseudo code

```
Input:  Input1:Input 14 sets of IQ data to be compressed;  InputWidth, OutputWidth: Selection of compression ratio;  AlgSelect: For the selection signals of the three algorithms, we encode the degree of parallelism of data.  Variables:  IQ:14 groups of IQ variables 1;  IQ:28 IQ exponents variables 1;  Exp:28 IQ exponents output;  Expout: Common exponent of block floating point output or IQ exponents output;  Parallel: Signals that control data parallelism;  OutputWidth: Selection of compression ratio;  AlgSelect: For the selection signals of the three algorithms, we encode the degree of parallelism of data.  Determine the degree of parallelism of data.  if (InputWidth==Exp) then  Calculate the bias and bias the input.  \end if
end function
function ExponentGenerate(IQ1): Lookup tables generate exponents.  return Exp1.  end function
function Round(IQ1, Exp1, OutputWidth, AlgSelect, Round):  if Round then  \begin if AlgSelect == 0 then  \begin Round(IQ1, Exp1, OutputWidth);  \end if  \end if
end function
function IQ_to_common(Mantissa, Exp):  if AlgSelect == 2 then  MantissaProcessing();  \end if
end function
```

**Fig. 2** Compression format requirements of ORAN in block floating point compression.
Algorithm 2 Decompress pseudo code

Input:
- Input: Input 14 sets of IQ data after compression;
- Exp: Common exponent of block floating point or IQ exponents of μ-law input;
- InputWidth: Output Width; Selection of compression ratio;
- AlgSelect: Provide a variety of algorithm options;
- BIAS: The Bias switch, 0 means off, 1 means on.

Output:
- Output: Output 14 sets of IQ data after decompression;

Variables:
- IQ: 14 groups of IQ variables;

function Daterecovery(Input, InputWidth, Exp, AlgSelect)
    Shift the data according to the exponent and the width of the input data to roughly restore the data.
    return IQ.
end function

data
function DPostprocess(IQ, AlgSelect, BIAS, Exp)
    Subtract the bias added during the compression phase.
    return Output.
end function

signal, we obtain the corresponding bias by looking up the table. Then we subtract the bias from the original signal and then compress it. The purpose of subtracting the bias is to move the original signal to an interval with a smaller quantization interval. Finally, we add the bias to restore the original signal when decompressing. In this way, the compression distortion can be significantly reduced.

In the case of using the μ-law in option 8 mode and 256QAM, the result with an output width of 13 bits is shown in Table I. Table I shows that the EVM treated with distortion minimization is about 1/6 of the EVM treated without distortion minimization. The distortion minimization methods designed in this paper greatly improved the compression accuracy [23].

| Table I | Functions of rounding and bias |
|---------|-------------------------------|
| EVM(%)  | Round and BIAS | Only Round | Only BIAS | None |
| 0.0134  | 0.0430          | 0.0206     | 0.0789     |

3.3 Hardware architecture design

Requirement on flexibility is high in base station. ASIP offers relatively higher flexibility than that of circuit configuration. The use of assembly coding tool could be easier than the use of module configuration tools. To adapt to ORAN, ASIP structure is used in our front-haul project. This ASIP is part of the front-haul processor. Because our instructions are the subset of the front-haul processor, further code optimization is limited by system constraint.

The hardware framework includes the control path and the data path, which will be integrated into the front-haul processor. Two main parts of the hardware can be found in the control path: the instruction flow controller and the instruction decoder [24]. The input of the control path includes the loaded instruction code from the program memory (PM), the flag from the data path, the branch target address, the configuration vector, and the asynchronous control request [25]. The output of the control path includes control signals to the data path, the address generator, and the control path itself, as well as immediate data from the decoded instructions [26]. The ASIP architecture and PC finite state machine are shown in Fig. 3 and Fig. 4.

Based on the algorithm, the functional modules were divided into several reusable modules and the execution phase in the architecture were divided into two steps: Execution1 and Execution2, as shown in Fig. 5 and Fig. 6. The discussed method improves the hardware efficiency.

3.4 Instruction set design

The front-haul data have different formats according to the configuration needs, which lead to changes in the parameters of compression [2]. The REPEAT instruction was required. These functions could be implemented with the three instructions in Fig. 7, Fig. 8, Fig. 9.

There is another compression method called A-Law in Table II, the algorithm is very similar to μ-Law and has the same precision. We thus will only discuss μ-law.

3.5 Hardware implementation

The silicon prove was based on SMIC 28nm digital CMOS technology. SRAM blocks were not included in the layout because it was only a part of the front-haul processor, and
Therefore, the front-haul module is sufficient to meet the needs of any high-end base station and can also be used for backbone Ethernet.

4. Discussions

4.1 Hardware overhead

The circuit speed and throughput of ASIC in this paper was based on static timing of annotated netlist from synopsys ICC. Following system design balancing, worst value appears seldom, it is accepted during compression, and be handled by FEC (forward error correction) through system design balancing.

From the results of IC layout, after the fusion of the three algorithms, the number of gates in the hardware modules of the multi-mode algorithm are 28457 cells, and the area is 100071 square microns. These show that the multi-mode compression and decompression algorithms are 72248 cells, and the area is 100071 square microns. The total number of compression / decompression module in this paper reduces the gate overhead by about 3/5 and the area by about 70%.

The compression / decompression module in this paper ran on a 1.2 GHz system, and could process 448 bits of data per clock so that the highest bandwidth could reach 537.6 Gbps. At present, the bandwidth of CPR1 / eCPR1 is usually 25 Gbps [27], which means that the bandwidth of the module designed in this paper sufficient for future uses. Therefore, the front-haul module is sufficient to meet the needs of any high-end base station and can also be used for backbone Ethernet.

4.2 Distortion measurement

The Error Vector Magnitude (EVM) is a measure of the difference between the equalized ideal symbol and the measured symbol, which is used to evaluate the distortion between the original data and the data processed by the compression algorithm [28]. EVM is an important index to measure the compression effect [29].

The formula of EVM is as follows:

\[ EVM(\% ) = \sqrt{\frac{E[|x - \bar{x}|^2]}{E[|x|^2]}} \times 100. \]  

where \( x \) is the original signal, \( \bar{x} \) is the decompressed signal.

In order to achieve an ergodic signal, we used the four data configurations (with different PRB numbers and modulation schemes) shown in Table III which were selected to simulate different downlink user scheduling signals.

Table IV: Simulation cases configuration

| Case index | PRB number | Modulation scheme |
|------------|------------|------------------|
| 1          | 100        | 64QAM            |
| 2          | 100        | 128QAM           |
| 3          | 100        | 256QAM           |
| 4          | 100        | 512QAM           |

Table V: EVM of IQ compression for front-haul option 8

| EVM(%)  | compression ratio |
|---------|------------------|
| 0.4716  | 46.88%           |
| 0.3016  | 51.34%           |
| 0.4075  | 57.59%           |
| 0.1526  | 75.00%           |
| 0.3050  | 46.88%           |
| 0.4014  | 57.59%           |
| 0.1513  | 75.00%           |
| 0.3078  | 46.88%           |
| 0.4010  | 57.59%           |
| 0.1518  | 75.00%           |

Table VI: EVM of μ-law compression for front-haul option 8

| EVM(%)  | compression ratio |
|---------|------------------|
| 0.4700  | 46.88%           |
| 0.3030  | 51.34%           |
| 0.4010  | 57.59%           |
| 0.1518  | 75.00%           |
| 0.3064  | 46.88%           |
| 0.4014  | 57.59%           |
| 0.1513  | 75.00%           |
| 0.3078  | 46.88%           |
| 0.4010  | 57.59%           |
| 0.1518  | 75.00%           |

Table VII: EVM of block floating point for front-haul option 7-2

| EVM(%)  | compression ratio |
|---------|------------------|
| 0.4700  | 46.88%           |
| 0.3030  | 51.34%           |
| 0.4010  | 57.59%           |
| 0.1518  | 75.00%           |
| 0.3064  | 46.88%           |
| 0.4014  | 57.59%           |
| 0.1513  | 75.00%           |
| 0.3078  | 46.88%           |
| 0.4010  | 57.59%           |
| 0.1518  | 75.00%           |
We can also see in Fig. 11 that the effect of the EVM measured in reference [18] is smaller than that in data and real IQ data sampled from front-haul. Therefore, special data set. The test vectors in this paper are random frequency domain of the downlink. Reference [18] uses a fix in the compression process. Although this can greatly remove the cyclic prefix in the reference. We can see that the compression performance of reference [18] is sufficient, yet it removes the cyclic prefix in the compression process. Although this can greatly improve the compression efficiency, it has some limitations in application. For example, it can only be used in the frequency domain of the downlink. Reference [18] uses a special data set. The test vectors in this paper are random data and real IQ data sampled from front-haul. Therefore, the EVM measured in reference [18] is smaller than that in this paper. We can also see in Fig. 11 that the effect of \( \mu \)-law compression in this paper is slightly better than that in reference [18]. IQ compression and BFP compression might not work well in frequency domain before FFT whitening. The compression performance of IQ compression and BFP compression can be better if a true front-haul data set is used. And the complexity of IQ compression method is the lowest among the three compression methods, 30% lower than that of \( \mu \)-law. Although its compression performance is not high, it can also meet our requirements for front-haul compression. One of the discoveries of the paper is that the EVM of \( \mu \)-law in this paper is much lower than that of other references. All compression modes meet the key requirements of CPRI data compression: EVM <3% [30].

### 5. Conclusion

In this paper, three compression algorithms were merged into an unified algorithm and implemented into one programmable and configurable module (an ASIP). To reduce data distortion, we introduced rounding, and we reduced the distortion of large numbers by automatic bias. A data compression instruction subset was defined and a pipelined hardware architecture was implemented accordingly. We minimized the delay through parallel processing and hardware loop instruction (REPEAT). The implemented ASIP reached 1.2 GHz while processing 448 bits in parallel, so that the maximum rate of data compression could reach 537.6 Gbit/s. Finally, compared with three single function modules, the cost of silicon is reduced to 3 / 5 through hardware reuse. The EVM as the measure of distortions was measured and compared. EVM of this paper is significantly lower than other references under the same conditions.

### References

[1] S.H. Kim, et al.: “Experimental demonstration of CPRI data compression based on partial bit sampling for mobile front-haul link in C-RAN,” OFC (2016) 1 (DOI: 10.1364/OFC.2016.W1H.5).

[2] CPRI Specification V7.0 (2015) https://www.cpri.info.

[3] eCPRI Specification V1.1 (2018) https://www.cpri.info.

[4] China Mobile Research Institute: “C-RAN: the road towards green RAN,” White paper version 1.0.0 (2018).

[5] T. Hu, et al.: “Method of I/Q data compression based on CPRI protocol,” Optical Communication Technology 7 (2015) 61 (DOI: 10.13921/j.cnki.issn1002-5561.2015.07.018).

[6] G. Bin, et al.: “Research on CPRI compression algorithm in LTE
and LTE-advanced systems,” Information and Communications Technologies (2013) 64.

[7] SG NR logical architecture and its functional splits, http://www.parallelwireless.com.

[8] L. Ramalho, et al.: “An LPC-based front-haul compression scheme,” IEEE Commun. Lett. 21 (2017) 318 (DOI: 10.1109/LCOMM.2016.2624296).

[9] C. Feng: “Data compression scheme of front-haul network based on LTE,” IJIEC 14 (2018) 153 (DOI: 10.3991/ijoe.v14i10.9312).

[10] ALTERA White Paper: “The emerging need for front-haul compression,” White paper (2016) https://www.thailand.intel.com.

[11] K.F. Nieman and B.L. Evans: “Time-domain compression of complex-baseband LTE signals for cloud radio access networks,” IEEE Global Conference on Signal and Information Processing (2013) 1198 (DOI: 10.1109/Globalsip.2013.6737122).

[12] R. Maiden: “Low-loss compression of CPRI baseband data,” (2014) https://www.edn.com/low-loss-compression-of-cpri-baseband-data/.

[13] Y. Ren, et al.: “A compression method for LTE-A signals transported in radio access networks,” ICT (2014) 293 (DOI: 10.1109/ICT.2014.6845126).

[14] D. Peng-ren and Z. Can: “Compressed transport of baseband signals in cloud radio access networks,” 9th International Conference on Communications and Networking in China (2014) 484 (DOI: 10.1109/CHINACOM.2014.7054343).

[15] H. Kwak, et al.: “Compression method to remove unnecessary MSBs of IQ data frames in C-RAN,” WWIC (2017) 315 (DOI: 10.1007/978-3-319-61382-6_26).

[16] B. Guo, et al.: “CPRI compression transport for LTE and LTE-A signal in C-RAN,” International 9th Conference on Communications and Networking in China (2013) 843 (DOI: 10.1109/ChinaCom.2012.6417602).

[17] B. Guo, et al.: “LTE/LTE-A signal compression on the CPRI interface,” Bell Labs Technical Journal 18 (2013) 117 (DOI: 10.1002/bltj.21608).

[18] J. Lee, et al.: “A simple and efficient IQ data compression method based on latency, EVM, and compression ratio analysis,” IEEE Access 7 (2019) 117436 (DOI: 10.1109/ACCESS.2019.2936218).

[19] International Telecommunication Union: “ITU-T Recommendation G.711,” (2000) https://www.itu.int/rec/T-REC-G.711

[20] O-RAN front-haul Working Group Control: User and synchroniza- tion plane specification (2019) https://www.o-ran.org/.

[21] C.W. Brokish: “A-law and mu-law companding implementation using the TMS320C54x,” Application Note (1997) SPRA163A.

[22] Z. Abraham: “Relative distance—an error measure in round-off error analysis,” Math Comp 39 (1982) 563 (DOI: 10.2307/2007334).

[23] J. Oh, et al.: “Low latency block scaling for baseband signal compression,” ICTC (2014) 105 (DOI: 10.1109/ICTC.2014.6983094).

[24] D. Liu: Embedded DSP Processor Design: Application Specific Instruction Set Processors (Morgan Kaufmann, Sweden, 2008) 1st ed. 30.

[25] Y. Huo and L. Dake: “High-throughput bit processor for cryptography, error correction, and error detection,” Microprocessors and Microsystems 61 (2018) 207 (DOI: 10.1016/j.micpro.2018.06.013).

[26] D. Liu: “ASIP (Application Specific Instruction-set Processors) design,” 2009 IEEE 8th International Conference on ASIC (2009) 16 (DOI: 10.1109/ASICON.2009.5351271).

[27] A. Agata, et al.: “Performance enhancement of optical access network in C-RAN using nonlinear quantization-based compression,” Opto-Electronics and Communication Conference and Australian Conference on Optical Fibre Technology (2014) 144.

[28] 3rd Generation Partnership Project (2019) https://www.3gpp.org/ ftp/Specs/archive/36_series/36.104/.

[29] J.-Y. Oh and H.-S. Chung: “Performance investigation of high compression data transmission based on group scaling for mobile front-haul,” ICTC (2016) 973 (DOI: 10.1109/ICTC.2016.7763344).

[30] K. Tanaka and A. Agata: “Next-generation optical access networks for C-RAN,” OFC (2015) 1 (DOI: 10.1364/OFC.2015.Th2E.1).