Optimized ARM-Based Implementation of Low Density Parity Check Code (LDPC) Decoder in China Digital Radio (CDR)

P. Vincy Priscilla[1], R. Padmavathi[2], S. Tamilselvan[3], Dr.S. Kanthamani[4]

[1,4] Department of ECE, Thiagarajar College of Engineering, Madurai- 625015, India.
[2,3] Jasmin Infotech Pvt Ltd, Chennai – 600100, India.

Abstract – *In recent days, traditional FM audio broadcasting is facing challenges from digital broadcasting technologies. There are several advanced digital audio broadcasting technologies like DAB, HD Radio, DRM, CDR, etc. Among them, China Digital Radio (CDR) is a Standard which allows to broadcast multiple audio streams. In the Channel Decoder part of CDR, the Low Density Parity Check (LDPC) codes are used for the efficient decoding of the received data. To decode the data received, LDPC codes uses various message-passing or iterative decoding algorithms. Though those algorithms can give desired results, the time taken to decode and its implementation complexity is more which is a major factor for real time application. Min-Sum algorithm reduces the overhead of dealing with complex mathematical and trigonometric expressions and proves to be efficient over other decoding algorithms. But it suffers from the slow rate of decoding and the low performance in terms of accuracy. These disadvantages are due to the increased number of iterations and implementation complexity. To optimize such difficulties, Layered Decoding algorithm is efficiently implemented in this paper. This approach provides the ability to retain and utilize previous decoded information and carry it forward to reduce the number of iterations required to decode and thereby improve the decoding convergence speed. Thus, by using layered LDPC decoding in Channel Decoder and implementing them in Jacinto Arm Cortex-A15 using NEON intrinsics, the number of cycles taken (MCPS) to decode the data is optimized in CDR.*

Keywords – China Digital Radio (CDR), Low Density Parity Check (LDPC) codes, Parallelism, NEON intrinsics

I.INTRODUCTION

China Digital Radio (CDR) is a standard which supports both hybrid mode and all-digital broadcasting mode for radio broadcasting in the FM band. It is a kind of IBOC (In-Band On-Channel) broadcasting scheme is a hybrid method for broadcasting digital radio and analog radio in the same frequency band simultaneously. It performs better by utilizing irregular spectrum allocation, LDPC coding schemes, time slicing, etc. Besides, CDR innovatively utilize the existing 200 KHz analog bandwidth to load both analog and digital radio signals without any spectrum change for the existing FM radio stations. The CDR standard allows to broadcast multiple audio streams and service information in one FM channel, as shown in Fig 1. In the CDR standard, LDPC coding is applied for main service data channel which is used for transmitting service multiplexing frames and channel logical frames.. The channel coding mechanism used for the main service data is LDPC coding with various suitable code rates. The LDPC block length is 9216 for all code rates and the corresponding coding delay is related to the transmission mode and modulation scheme. Here, a code rate of ½ is used with Quadrature Phase Shift Keying (QPSK) Modulation.

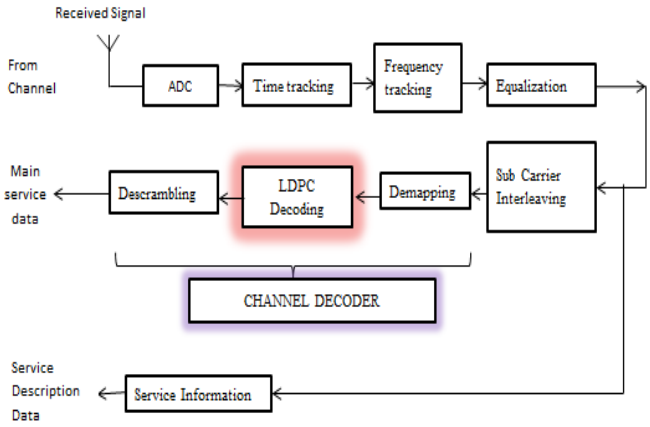

Fig 1: Block diagram of CDR Receiver chain

II. OVERVIEW OF LOW DENSITY PARITY CHECK (LDPC) CODES & JACINTO ARM CORTEX A15

LDPC is a Forward error-Correction (FEC) code. Forward error-Correction is a technique used for error control in data transmission over unreliable or noisy channel. LDPC codes are block codes with parity-check matrices (H) that contain only a very small number of non-zero entries. It is the sparseness of Parity Check Matrix which guarantees both a decoding complexity which increases only linearly with the code length and a minimum distance which also increases linearly with the code length. LDPC is a linear block code whose parity check matrix is sparse i.e it contains only few number of 1s per row or column. Block codes can be successfully used with the LDPC iterative decoding

algorithms if they can be represented by a sparse parity-check matrix. LDPC codes do not show an error floor at high SNRs as they are long and very sparse codes. The implementation complexity is low in encoding and decoding algorithms due to *sparse parity check matrix of LDPC.* Also as LDPC is a Forward Error-correction code, it provides high efficiency and is reliable for information transfer (Over Bandwidth). Here, Irregular LDPC codes are used. LDPC codes are decoded iteratively using a graphical representation of their parity-check matrix. Tanner Graph is the graphical representation of Parity-check matrix of LDPC codes as shown in fig. 2. In the tanner graph, Check nodes denotes to Number of Parity-check equations and Bit nodes denotes to Number of bits in a code word.

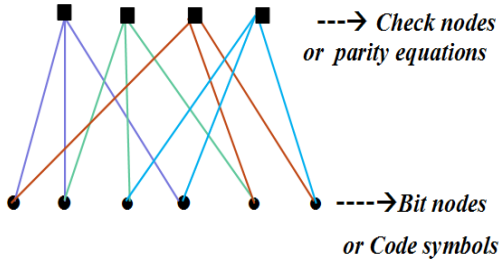

Fig 2: Tanner Graph

JACINTO ARM CORTEX A-15: The LDPC decoder is to be implemented in a high performance processor. Here, Jacinto ARM Cortex-A15 is the ARM Processor used for its implementation. It is an Advanced RISC Machine (ARM) which uses the RISC architecture. It provides the possibility of Single cycle execution and pipelined execution which helps in optimization. The advantages of ARM processors includes Auto-increment and auto-decrement addressing modes to *optimize* program loops and could handle instructions that combine a shift with an arithmetic or logical operation. It uses Load/Store architecture and is a 128 bit processor. ARM cortex A15 incorporates Big LITTLE technology (i.e) it uses LITTLE processors which are designed for high power efficiency whereas Big processors are designed for high performance (high computation speed). Its applications includes better audio playback. It is a first generation processor with ARM v7 debug architecture. ARM supports various instruction sets architecture extensions such as Jazelle, Thumb, Thumb EE and NEON.

ARM NEON: ARM NEON technology is an advanced SIMD (Single Instruction Multiple Data) Architecture extension for the ARM Cortex-A series and Cortex– R52 processors. It is intended to improve the multimedia user experience by accelerating audio encoding/ decoding. NEON accelerates signal processing algorithms and functions to speed up applications such as audio and video processing.

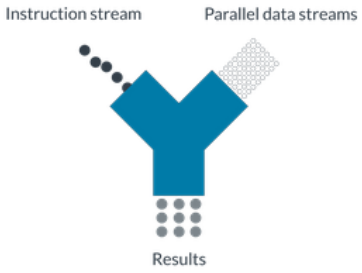

Fig 3: SIMD Architecture

NEON registers are considered as vectors of elements of the same datatype. Multiple datatype are supported by the technology. ARMv7-A supports both floating and integer datatypes. The NEON instructions perform the same operations in all lanes of the vectors. The implementation in NEON technology can also support issue of multiple instructions in parallel which is shown in fig 3 and fig 4.

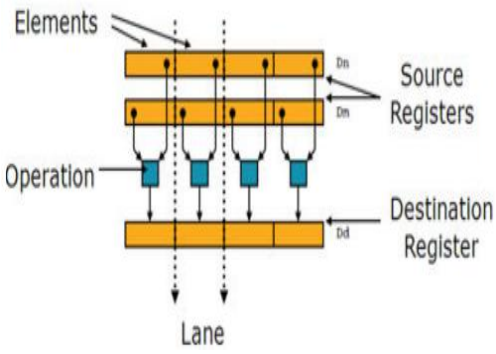

Fig 4: Parallelism in NEON

The goal of NEON is to provide a powerful, yet comparatively easy to program SIMD instruction set that covers integer data types of up to 64 bit width as well as single precision floating point (32 bit). NEON cannot be part of the ALU pipeline of the processor. Instead it shares its sixteen 128 bit registers with the vector floating point unit. NEON can be used in multiple ways such as NEON enabled libraries, compiler's auto-vectorization feature and NEON intrinsics. Auto-vectorization feature is supported by ARM compilers wherein they exploit NEON functionality automatically. These features of NEON are used in the decoding part of CDR to achieve better optimization.

III.ALGORITHM LEVEL OPTIMIZATION

Various message-passing or iterative decoding algorithms are available to decode the received data.

- **Binary Erasure Decoding**
  It introduces more errors into the received codeword in the binary symmetric or AWGN channels

- **Bit flipping algorithm**
  This results in the formation of Cycles in the Tanner graph.

- **Sum Product decoding**
  The accuracy of decoding is high. But the implementation complexity is also high due to large number of iterations.

- **Layered Decoding algorithm**
  The above disadvantages are overcome by efficiently implementing Layered Decoding of LDPC

LAYERED DECODING ALGORITHM: Another approach towards iterative decoding is based upon decoding in layers. The parity check matrix can be viewed as the horizontal layers and each layer can represent a component code. The intersection of all these codes (layers) forms the full code. Some form of soft-input soft-output (SISO) decoding can be applied to each layer in sequence. As each next layer starts decoding, its inputs are combined from the channel inputs and the extrinsic probability outputs from the SISO decoding on the last layer processed, or other prior layers if necessary. This overall process is repeated for several iterations. Iterations within a layer via the SISO decoder can be called the sub-iterations and the overall process repetitions labeled as super – iterations. Figure 5 shows this layered decoding process.

In layered decoding algorithm, the parity-check matrix H is divided into several layers based on rows. The updated check node information of top layer can be applied immediately to calculate the check node information of the next layer. Bit nodes posteriori probability is updated after the information from check node to bit node updates. Due to update the posteriori probability frequently, the convergence speed becomes higher. In this approach, portions of the most recently updated R_{mj} terms are more quickly utilized again in the algorithm. Assume that

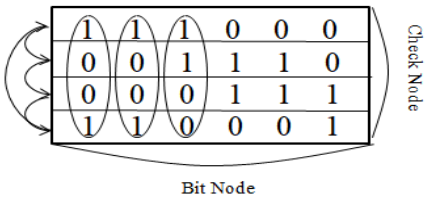

Fig 5: Layered Decoding Process

the rows are grouped into non-overlapping subsets where each subset has the following property. Each column in the parity check matrix for this subset portion has at most a weight of one i.e., at most one entry. Each subset is processed as a unit, one subset after another. First initialize $L(q_j), \forall j$ to the received soft data; typically $-2r_j/\sigma^2$ for a simple modulation on AWGN channels. Repeat the following for each subset of rows (and for each bit node j) as shown in fig 6. For all m in subset k of the rows:

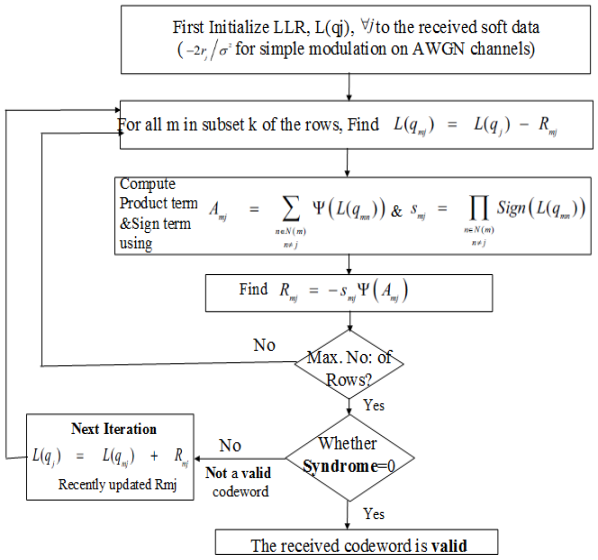

Fig 6: Steps for Layered Decoding

Note that the R_{mj} term is the most recently updated whereas R_{mj} from previous equation uses the value held upon starting this subset of rows.

The Layered architecture can be viewed in two ways. One, the decoding throughput can be doubled with less memory. Or two, by keeping decoding throughput constant, a decoder can be built of about half the size of the previous architectures. This can be achieved by cutting in half the parallelization factor of the decoder architecture. This cuts the size of the logic portion almost in half. These factors provides a greater advantage that either throughput or the memory size can be varied based on the requirement. Thus, in the Layered decoding of LDPC, the bit node update is done after each layer of the Parity Check Matrix. So, within a single iteration the bitnode probabilities are updated immediately and the syndrome is checked. Thus, by performing decoding for each layer, the number of iterations required for decoding is reduced also the execution time is reduced significantly. Hence, it is well suited for real time application. The advantages of layered decoding are:

- The faster decoding is done since the number of iterations is reduced and
- A significant reduction in memory size.

IV.ARCHITECTURE LEVEL OPTIMIZATION

The Layered LDPC decoding is applied to the channel decoder of CDR is implemented in the ARM Cortex A15 processor. The LDPC coding is compiled using the ARM tool chain which holds the libraries for NEON intrinsics.

OPTIMIZATION USING NEON INTRINSICS: The various NEON instructions of ARM Cortex A15 are listed below:

a) NEON intrinsic to load multiple single elements: This instruction loads elements from memory into one, two, three, or four registers, without de-interleaving. Every element of each register is loaded. Example: *uint16x4_t vld1_u16(__transfersize(4) uint16_t const * ptr);*

b) NEON intrinsic to convert floating point to fixed point: This instruction converts a value in a register from floating-point to fixed-point, or from fixed-point to floating-point. Software can specify the fixed-point value as either signed or unsigned. The floating-point value can be single-precision or double- precision. The fixed-point value can be 16-bit or 32-bit. Conversions from fixed-point values take their operand from the low-order bits of the source register and ignore any remaining bits. Example: *int32x2_t vcvt_s32_f32(float32x2_t a);*

The level of parallelism that can be achieved depends on the number of bits that is used in the intrinsics such as 32x2 & 32x4. As the number of bits accessed in a particular intrinsic is more, the amount of parallelism obtained is also high.

Similarly, different NEON intrinsics are used to achieve better optimization.

V. RESULTS AND DISCUSSIONS

On implementing the LDPC decoding of CDR in Jacinto ARM Cortex A15, the number of MCPS taken by LDPC decoder block of CDR was 93 MCPS and the throughput obtained was 3.3191 Mbps. This is due to the usage of Sum Product Algorithm which results in more number of iterations. Whereas on using layered LDPC decoding, the number of iterations taken is reduced to 62 MCPS. Similarly, while using NEON intrinsics of the ARM processor to the decoder part of CDR, the MCPS taken by the decoder is reduced since NEON uses parallelism and scheduling for the better performance of the arithmetic and logical operations. The value of the various parameters analyzed before and after optimizations are listed in the table 1.

PARAMETERS	BEFORE OPTIMIZATION	AFTER OPTIMIZATION
Million Cycles Per Second (MCPS)	93	63
Throughput (in Mbps)	3.32	4.23
Memory Occupied (in MB)	0.54	0.54

Table 1: Values of the parameters obtained before and after optimization

Thus, after the implementation of layered decoding of LDPC along with NEON intrinsics to the decoder of CDR, the MCPS is reduced which in turn increases the throughput. Thus, the optimization in terms of MCPS and throughput is achieved by using layered decoding of LDPC and by the parallelism provided by the NEON instruction sets of Jacinto ARM Cortex A15.

VI. CONCLUSION

Thus, the idea of optimizing the channel decoder of CDR is done by using the Layered decoding of LDPC codes and also by the NEON optimization in ARM processor. The implementation complexity of the Layered decoding of LDPC is lesser when compared to the various message passing algorithms of LDPC used earlier such as the Sum-Product decoding and Min-sum Algorithm. This reduction in implementation complexity is due to the immediate bit node update after the completion of each layer which in turn reduces the number of iterations. Also, the efficient usage of NEON instruction set provides the sufficient parallelism in performing the various arithmetic and logical operations which helps in optimization. The corresponding MCPS, throughput and Memory occupied before and after optimization of Decoder have been measured. Thus, the optimization of LDPC decoding in ARM with low MCPS (Million Cycles Per Second) and memory is achieved. The future work can be extended by making modifications in the usage of the algorithm such as varying the number of layers used, etc. to achieve better optimization in decoding.

VII. REFERENCES

1. Yun Wang, “Software-Defined Radio Receiver Design And Development For China Digital Radio (CDR)”, 2015.

2. Jia. Li, Gaigai. Yang, Zhiqiang. Zhao, “An Improved-Performance Decoding Algorithm of LDPC Codes for Layered Decoding” , IEEE International Conference on Communication Problem-solving (2014).

3. ARM DUI 0491E Compiler toolchain Version 5.0 - Compiler Reference, ARM copyrights, 2010-2011.

4. Mohammad Rakibul Islam, Dewan Siam Shafiullah, Muhammad Mostafa Amir Faisal, Imran Rahman, "Optimized Min-Sum Decoding Algorithm for Low Density Parity Check Codes", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No. 12, 2011.

5. Sarah J. Johnson,"Introducing Low-Density Parity-Check Codes", 2010.

6. Dale E. Hocevar", A Reduced Complexity Decoder Architecture via Layered Decoding for LDPC Codes", 2004.

7. Amin Shokrollahi, "LDPC Codes: An Introduction", 2003.

8. R. G. Gallager, "Low-density parity check codes," IRE Trans. on Information Theory, vol. IT-8, pp.21-28, Jan. 1962.