Soft-decision Forward Error Correction in a 40-nm ASIC for 100-Gbps OTN Applications

Sameep Dave, Lawrence Esker, Fan Mo, William Thesling, James Keszenheimer, Russell Fuerst
ViaSat, Inc
4830 East 49th Street, Cuyahoga Heights OH 44125
Tel: (216) 706-7800, Fax: (216) 706-7801
Sameep.Dave@viasat.com

Abstract: Soft-decision forward error correction provides high coding gain, but typically with high complexity. We present a soft-decision turbo product code with >11 dB NECG and reasonable complexity in a 40-nm ASIC for 100-Gbps applications.
OCIS codes: 060.4510

1. Introduction

Ever increasing demands for data and bandwidth have fueled the need to increase the single-channel dense wavelength division multiplexing (DWDM) capacity from 10 Gbps to 40 and 100 Gbps, but with the same channel bandwidth. Forward error correction (FEC) is considered an attractive, cost-effective solution for gaining the additional sensitivity that is required at higher data rates. FEC exploits redundant information added at the transmitter during encoding to help recover the signal on the receive side. The performance gain provided by FEC enables extending the network reach by increasing the maximum span length, reducing the optical transmit power required to close the link, enabling an increased number of channels in a DWDM system, and providing the option of using lower-quality, lower-cost components to meet the same system requirements.

There are a variety of advanced FEC codes and decoding algorithms, each of which achieves different coding gain performance. These include convolutional turbo codes (CTCs) [1], low density parity check codes (LDPCs) [1], turbo product codes (TPCs) [2-4], and a variety of concatenated codes. These codes rely on iterative decoding algorithms to help recursively correct errors in the received data streams and can get to within 1 dB of the theoretical performance bounds. The maximum coding gain that can be achieved also depends on if the decoder can be provided hard-decision or soft-decision inputs. In the case of hard-decision inputs, the decoder receives a stream of '1's and '0's, while in the soft-decision case, the decoder receives an additional metric that describes the reliability of each bit being a '1' or a '0'. Soft-decision decoding leads to higher coding gain performance compared to hard-decision decoding, but at the expense of higher implementation complexity.

Given the high degree of parallel processing required to achieve throughput in excess of 100 Gbps, the complexity of soft-decision based FEC can get prohibitive. The authors analyzed the various classes of advanced FECs with regards to their performance and implementation complexity for 100-Gbps systems and found that TPCs are advantageous for the following reasons:

- Very high coding gain
- High and deterministic minimum distance / excellent asymptotic performance / no error floors down to 10^{-15} operational BERs
- Faster convergence / lesser decoding iterations leading to lower complexity and lower latency
- Higher tolerance to soft-input resolution so few bits of ADC input resolution are required

In this paper the performance and complexity results for a soft-decision TPC decoder designed for a 40-nm ASIC are presented. A brief introduction to TPCs is presented in Section 2, followed by the decoding architecture in Section 3, and finally the performance and complexity of the decoder in Section 4.

2. TPCs

TPCs involve serial concatenation of two or more Bose-Chaudhuri-Hocquenghem (BCH) coders separated by a simple row/column interleaver. This gives them a product code structure as shown in Fig. 1. Every row in Fig. 1 is an (n1,k1) BCH code where k1 is the number of information bits and n1 is the number of coded bits (information + check/parity bits). Similarly, every column in Fig. 1 is an (n2,k2) BCH code where k2 is the number of information bits and n2 is the number of coded bits (information + check/parity bits). The overall TPC code has an information blocksize of k1*k2 and coded blocksize of n1*n2 for a code rate of (k1*k2)/(n1*n2) and an overhead of (((n1*n2)/(k1*k2)) – 1)*100 percent.

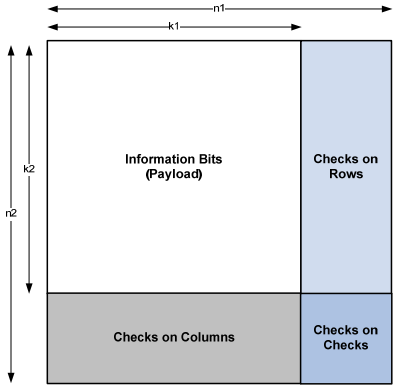

Fig. 1. TPC code structure.

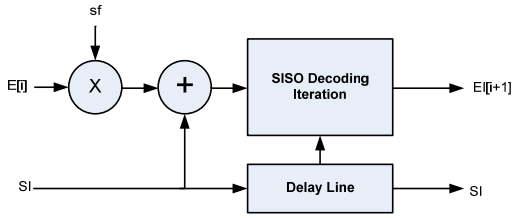

Fig. 2. TPC SISO decoding iteration.

The decoding procedure for TPCs is similar to the decoding of other product codes. Decoding can be performed either row-wise first and then column-wise, or vice-versa. For iterative decoding, the decoding process at each step should be soft-input soft-output (SISO). One row-wise decoding followed by one column-wise decoding constitutes a single iteration. A schematic of the elementary row or column decoder at any stage of an iteration is shown in Fig. 2.

The soft-input data [SI(i)] for the i^{th} decoding iteration is given by:

$$[SI(i)] = [SI] + sf * [E(i)], \quad (1)$$

where [SI] is the received data corresponding to the transmitted codeword, [E(i)] is the extrinsic information computed by the previous decoder iteration, and *sf* is the scaling factor. The elementary decoder calculates the soft-output [SO(i)] and delivers at its output the updated extrinsic information:

$$[E(i+1)] = [SO(i)] – [SI(i)]. \quad (2)$$

3. Decoding Architecture

The top-level block diagram of a TPC decoder includes a deinterleaver, a pre-processor, SISO iterations, and a post-processor. The deinterleaver parses data into TPC blocks if interleaving is used for additional burst error protection. A pre-processor arranges the data in a suitable format to facilitate highly parallel decoding. Each SISO iteration performs SISO decoding, and a post-processor removes parity bits and passes the corrected information bits.

Performance simulations of the TPC decoder showed that four decoding iterations were sufficient to extract most of the coding gain from the TPC code. Additional iterations do not provide sufficiently more gain to justify the extra complexity. The RTL design uses 4 copies of the SISO decoding engine, which are kept identical for the benefit of synthesis, physical layout, and other back-end work. This pipelined architecture for the iterations makes the design flow in a forward only direction, which also helps physical layout and routing.

4. Performance Results

The TPC decoder performance was simulated using fixed-point, C-based code that was proven to be bit-true under a variety of random stimuli to the RTL decoder. Because of the speed limitation of the C software, an FPGA development platform was used to run the RTL decoder at a higher speed. Fig. 3 shows the bit error rate (BER) performance for a ~15% overhead TPC decoder in an additive white Gaussian noise (AWGN) channel with QPSK modulation (assuming an ideal demodulator). As seen from Fig. 3, this code can achieve a net electrical coding gain (NECG) slightly higher than 11 dB for a 10^{-15} BER.

The performance of the 15% overhead TPC was also evaluated with differentially encoded QPSK (DQPSK) in an AWGN channel and also in the presence of miscellaneous optical fiber impairments (e.g., dispersion) using a reference digital fixed-point demodulator design that included equalization. The BER_{in} versus BER_{out} results are presented in Fig. 4, which indicate that the excellent correction performance of the TPC codes in AWGN channels is readily extended to optical communication channels.

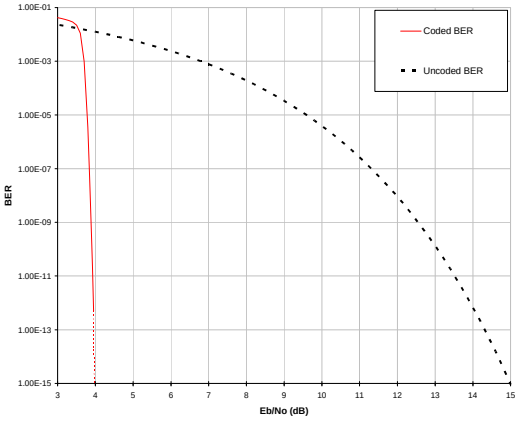

Fig. 3. TPC decoder performance in an AWGN channel with coherent QPSK.

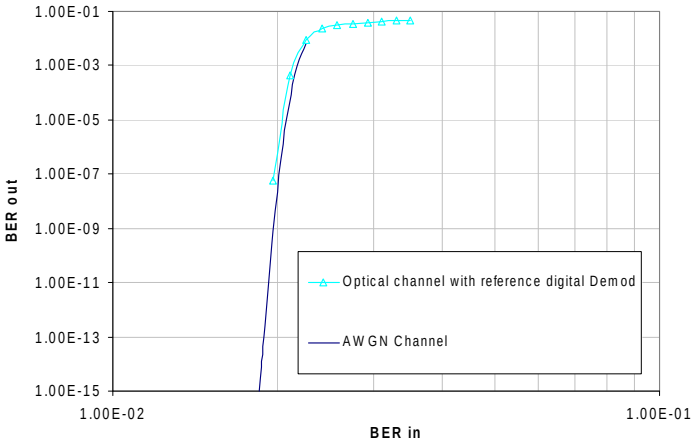

Fig. 4. TPC decoder performance with differentially coded QPSK, shown for an AWGN and an optical channel.

The RTL design was synthesized using the Synopsys design compiler and targeted towards a 40-nm technology ASIC. The design meets the desired 500-MHz timing with sufficient margin. The synthesis results for the design are presented in Table 1.

Table 1. Synthesis results for a 40-nm ASIC

Complexity Metric	**Approximate Values**
Flip-Flops	640 k
Combinatorial Gates	6.3 M
RAM	9 M bits
Area	23 sq. mm
Power	7 W

The results of this effort indicate that high coding gains offered by soft-decision decoding can be used for 100-Gbps OTN applications. TPCs provide an attractive option for such applications with reasonable complexity.

5. References

[1] Shu Lin, and Daniel J. Costello, Jr., “Error Control Coding: Fundamentals and Applications,” Prentice Hall, NJ, 2004.

[2] Ramesh Mahendra Pyndiah, “Iterative Decoding of Product Codes: Block Turbo Codes,” International Symposium on Turbo Codes, Brest, France, pp. 71-79, Sept. 1997.

[3] Ramesh Mahendra Pyndiah, “Near-optimum Decoding of Product Codes: Block Turbo Codes,” IEEE Transactions on Communications, Vol. 46, No. 8, Aug. 1998, pp. 1003-1010.

[4] S. Dave, J.Kim and S.C. Kwatra, “An Efficient Decoding Algorithm for Block Turbo Codes,” in IEEE Transactions on Communications, Vol. 49, No. 1, Jan. 2001, pp. 41-46.