

A Bit-Serial Approximate Min-Sum LDPC Decoder and FPGA Implementation

Ahmad Darabiha, Anthony Chan Carusone,
Frank R. Kschischang

University of Toronto, Toronto, Canada
ISCAS, Kos, Greece
May 2006

Outline

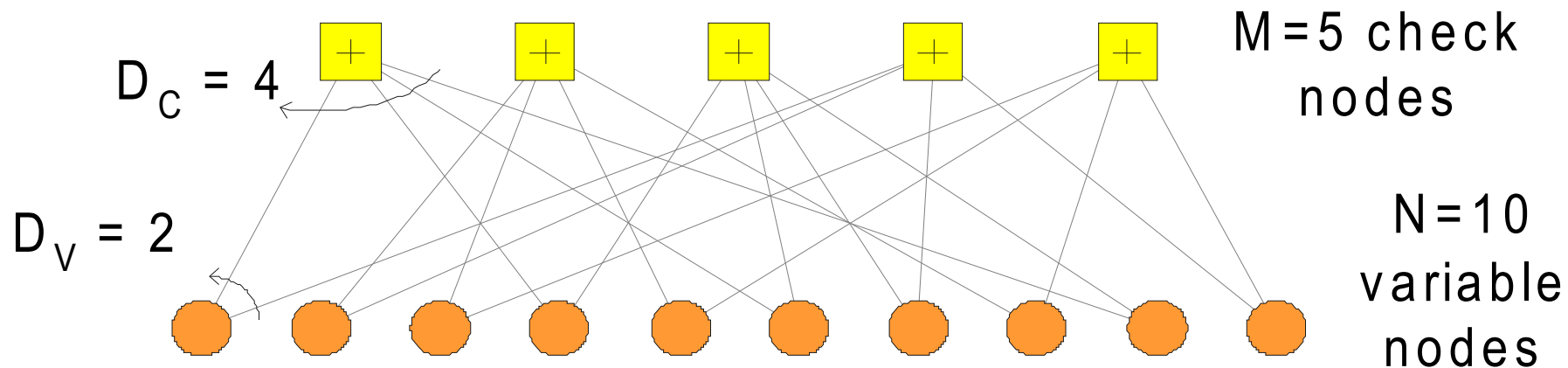
- Introduction to LDPC codes/decoders
- Proposed techniques
 - Bit-serial message passing
 - Approximate Min-Sum decoding
- FPGA implementation
- Conclusion

Introduction

- Low-Density Parity-Check (LDPC) codes
 - A sub-class of **Error Control Codes** (ECC)
 - Perform better than Turbo and Reed-Solomon codes
 - Approach Shannon limit

- LDPC codes are adopted for
 - IEEE 802.3 10Gbit Ethernet standard
 - Input BER $> 10^{-3}$
 - Output BER $< 10^{-13}$
 - Code rate 0.84
 - DVB-S2 Digital Video Broadcast standard

LDPC Codes: Structure



$$H_{M \times N} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

- Each bit participates in D_v parity checks
- Each check consists of D_c bits
- Good LDPC codes are **long** (large N) and with a **random-like** graph

Min-Sum LDPC Decoding

- A form of iterative message passing decoding
- Messages in Log-Likelihood Ratio (LLR)
 - $\text{Log} (P (x=0) / P (x=1))$
- Each iteration does two updates:

- Check node update

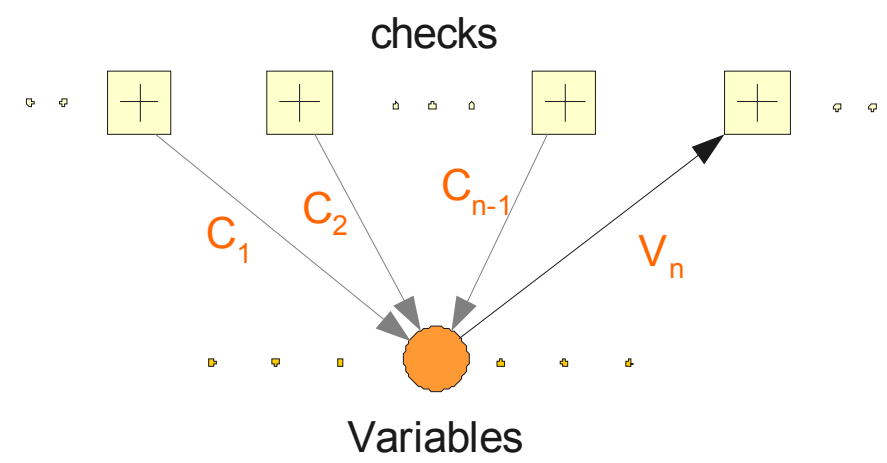
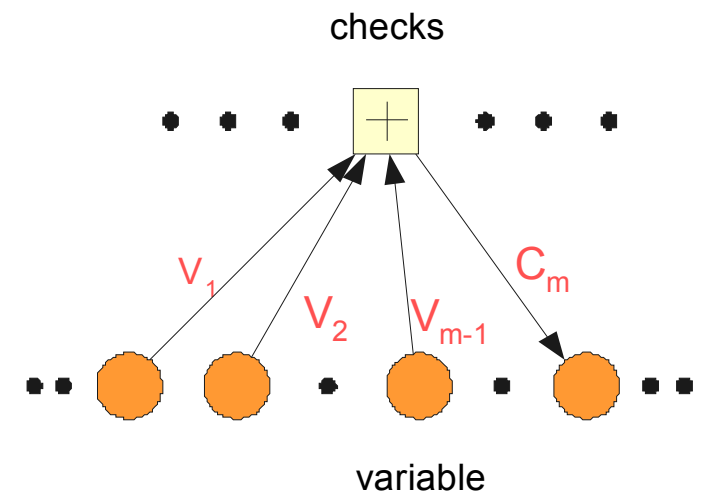
$$c_m = \text{chk}(v_1, \dots, v_{m-1})$$

$$= (\text{sgn}(v_1) \dots \text{sgn}(v_{m-1})) \text{Min}(|v_1|, \dots, |v_{m-1}|)$$

- Variable node update

$$v_n = \text{var}(c_1, c_2, \dots, c_{n-1})$$

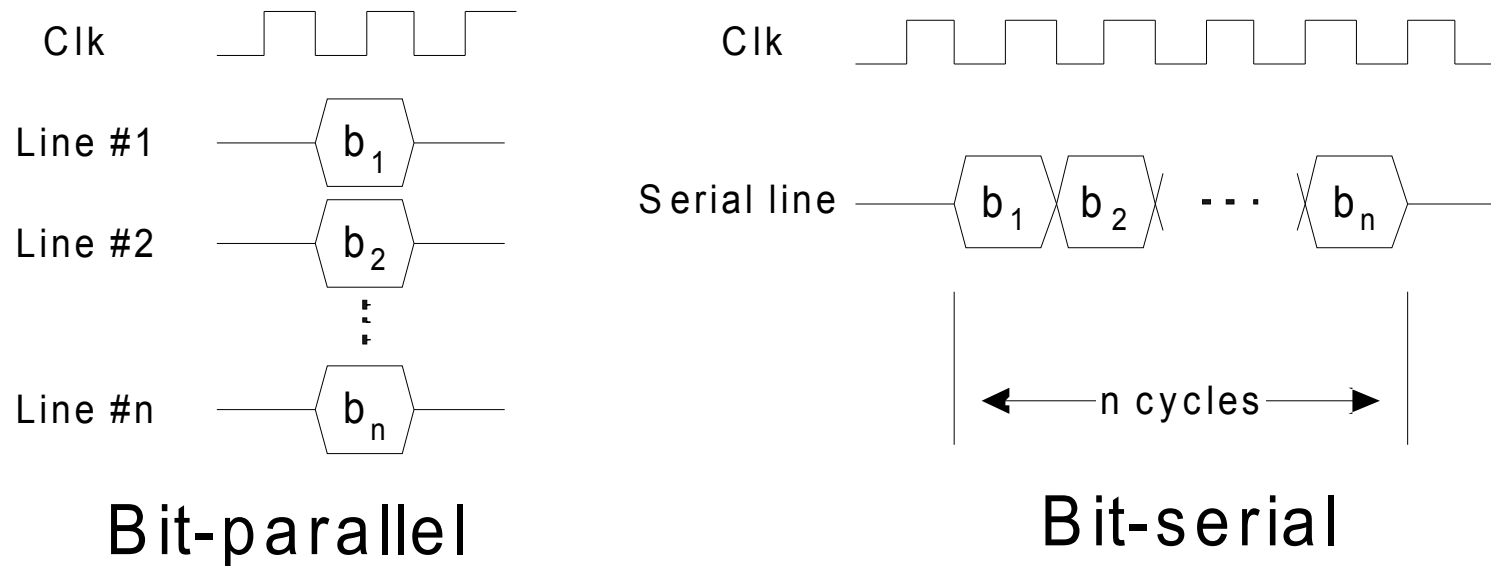
$$= c_1 + c_2 + \dots + c_{n-1}$$



LDPC Decoders: Architecture

- We use fully-parallel architecture
 - Allows high throughput
 - Graph is directly mapped to hardware
- Major challenge:
 - Complex interconnection
 - Wire capacitance
 - Power dissipation
 - Wire delay
 - Routing congestion
 - Larger area
- We propose a **bit-serial** scheme to reduce interconnections

Bit-Serial Message-Passing



- bit-serial scheme
 - transfers an n -bit message in n clock cycles over a single wire
 - is Ideal for Min-Sum decoding
 - reduces the number of wires => reduced routing congestion
 - both `Min` and `Sum` functions are naturally bit-serial
 - facilitates efficient gear shift decoding

Min-Sum Approximation

Approximation to Min-Sum Decoding

- In original Min-Sum each check node needs to find first and second minimum

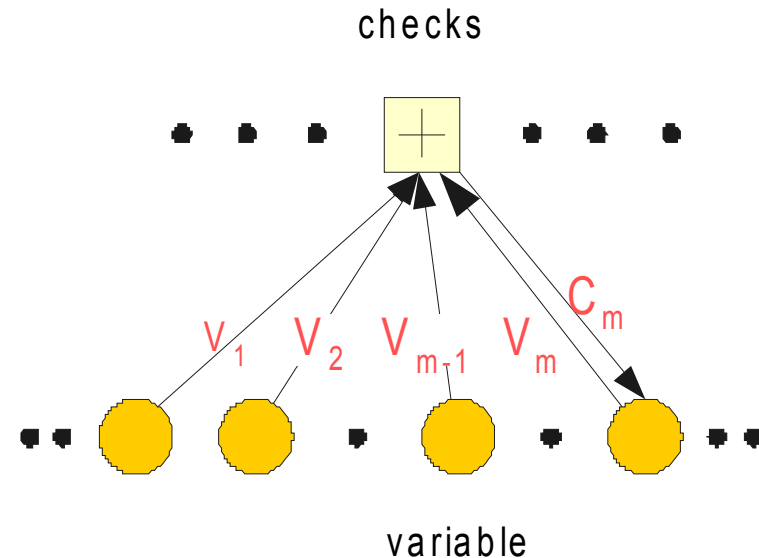
- We approximate Min-Sum:

- Variable node the same as conventional Min-Sum
- Check node update:

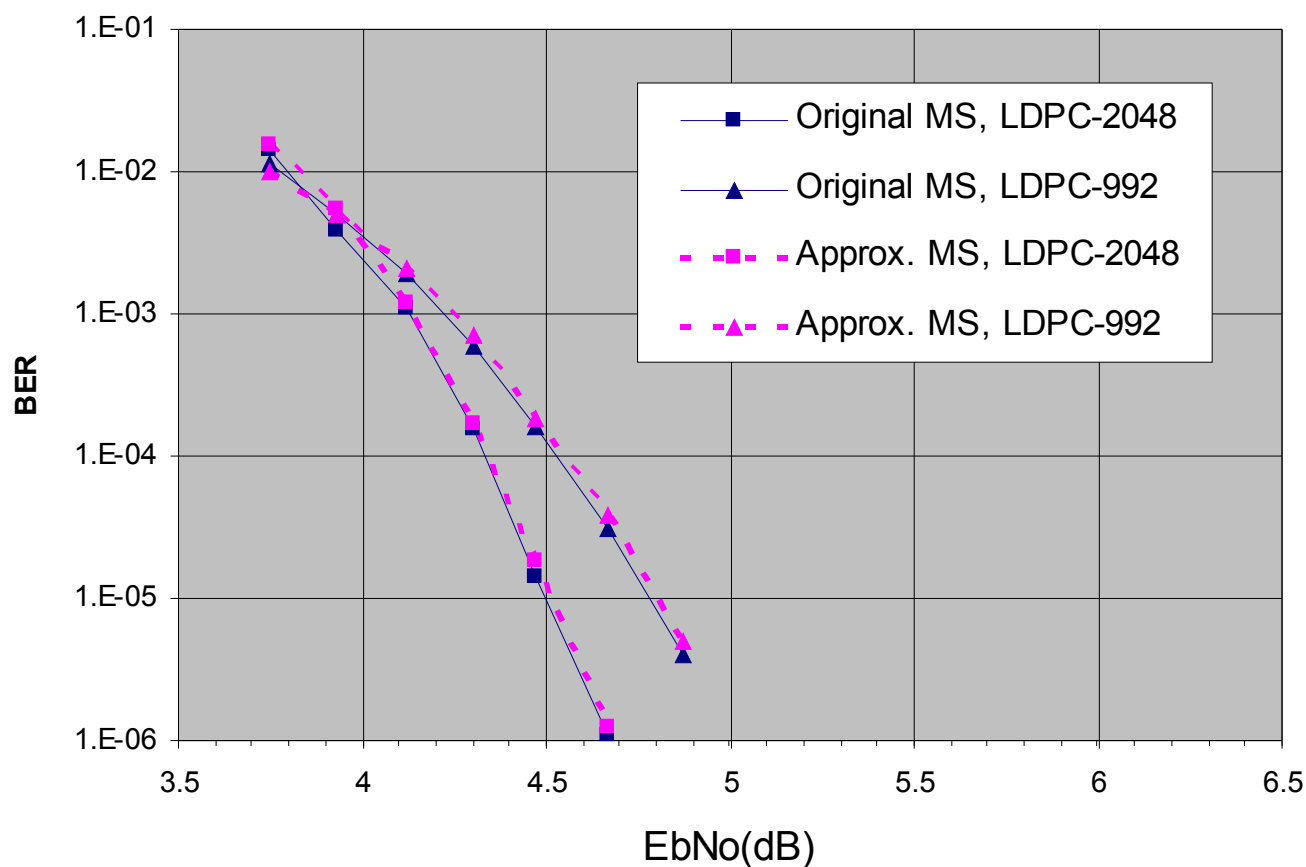
$$c_m = \text{check}(v_1, \dots, v_m)$$

$$= (\text{sgn}(v_1) \dots \text{sgn}(v_{m-1})) \text{Min}(|v_1|, \dots, |v_m|)$$

- In Approximate Min-Sum **only the first minimum** needs to be found
 - reduces the check node logic by about 50%

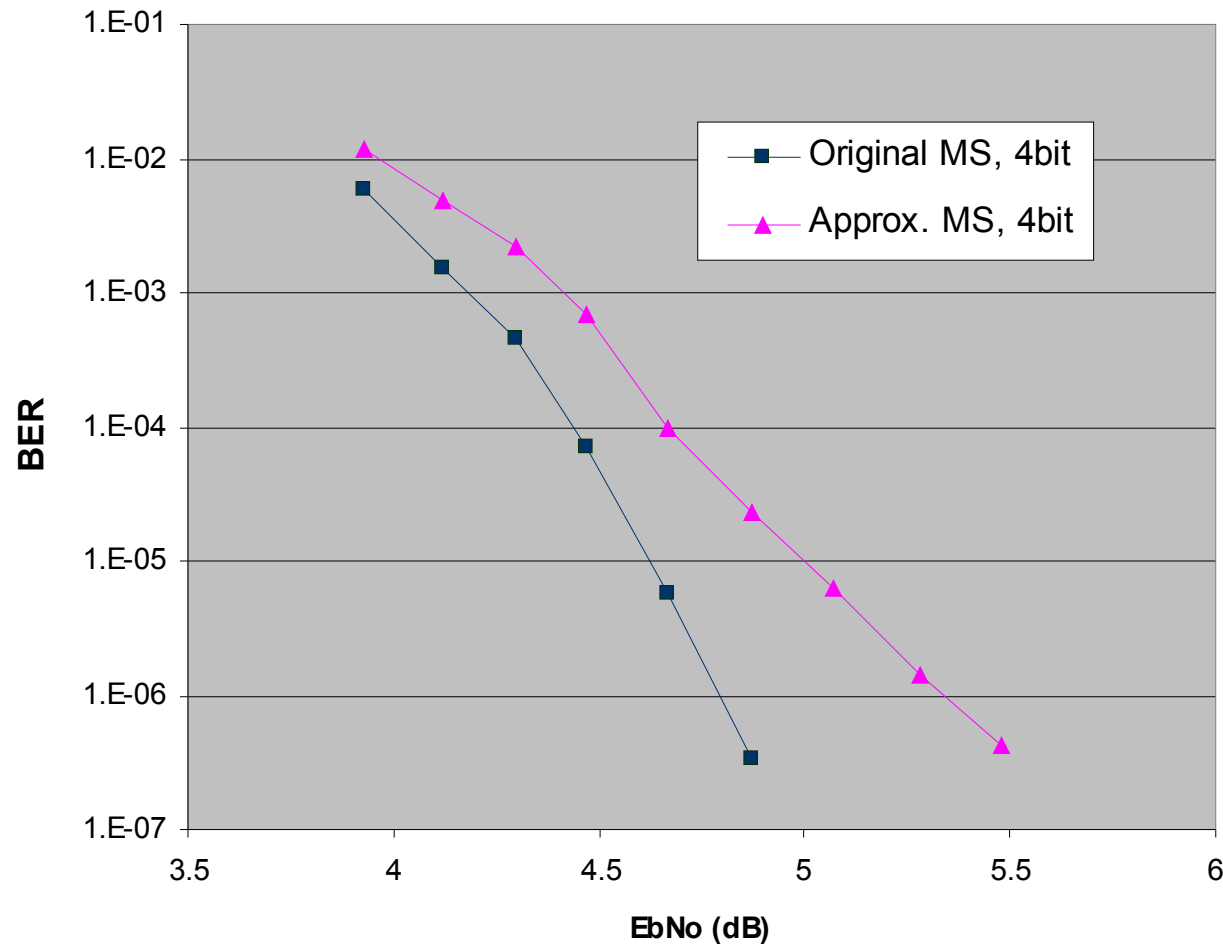


Approximate Min-Sum: Performance (Full-Precision)



Under full-precision computations, conventional Min-Sum and approximate Min-Sum perform closely

Approximate Min-Sum: Performance (Quantized)

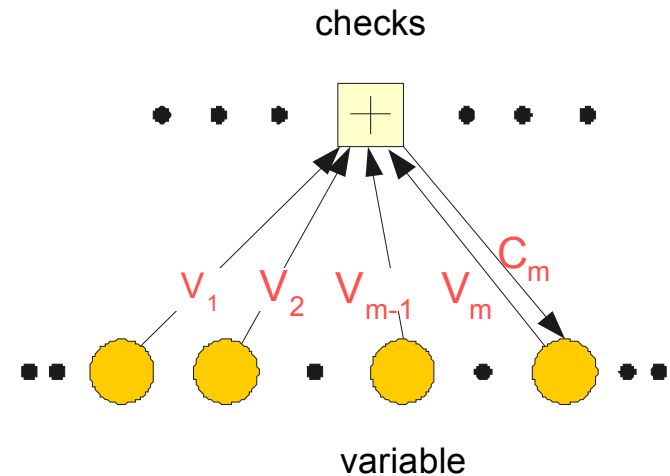


With quantized calculations, there is more than 0.5dB difference in performance between conventional and approximate Min-Sum

Approximate Min-Sum + Correction

- We add a correction factor to the check update rule to reduce the gap between conventional Min-Sum and approximate Min-Sum
- Variable node unchanged
- Check node:

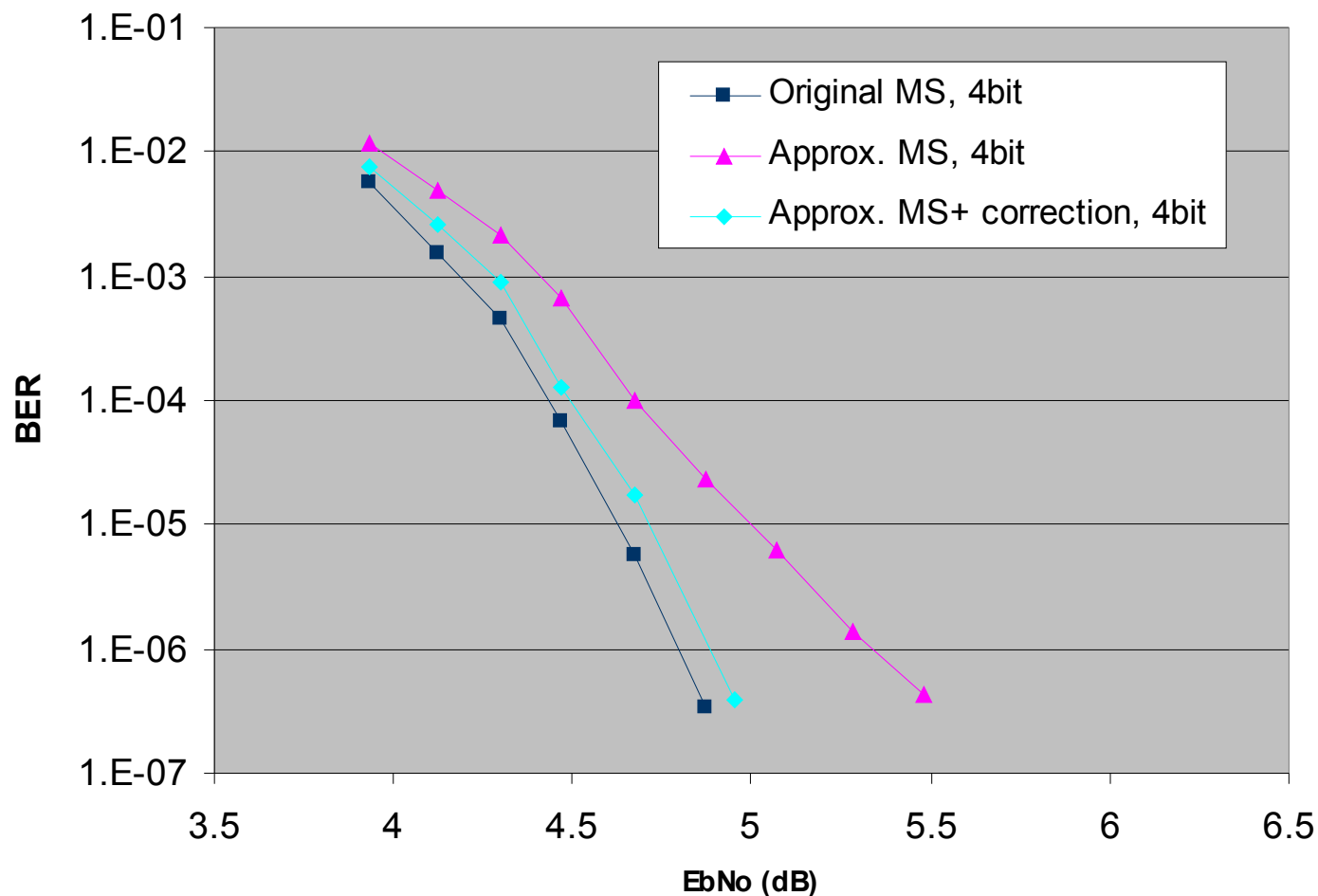
- Define $M = \min(|v_1|, \dots, |v_{m-1}|, |v_m|)$
- Define $T = \text{No. of identical absolute minimums}$



- $C_m = \text{check}(v_1, v_2, \dots, v_m)$

$$= \begin{cases} (\text{sgn}(v_1) \dots \text{sgn}(v_{m-1})) M + 1 & \text{if } (V_m = M \ \& \ T = 1), \\ (\text{sgn}(v_1) \dots \text{sgn}(v_{m-1})) M & \text{otherwise} \end{cases}$$

Approximate Min-Sum + Correction: Performance



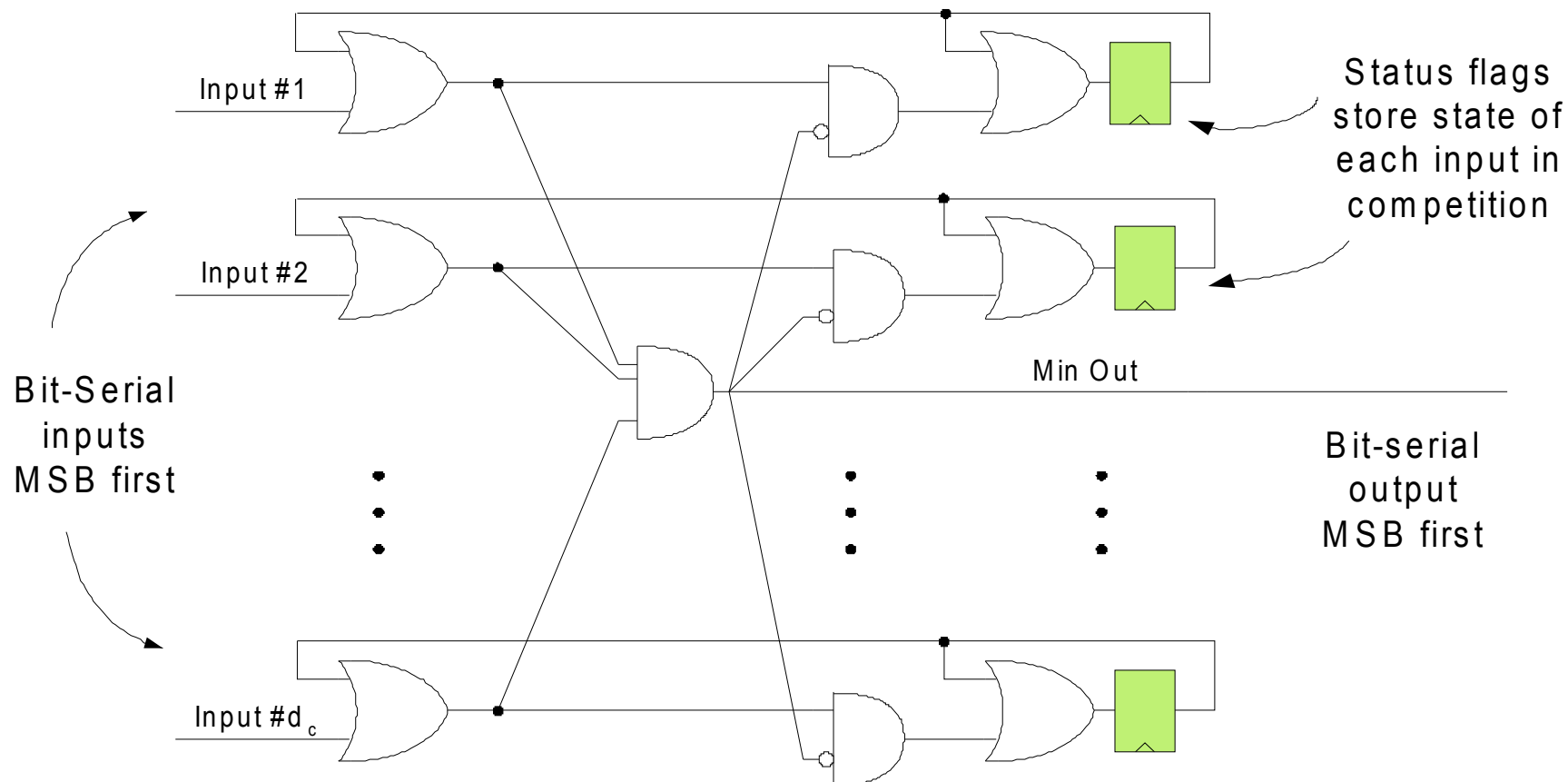
By adding the correction factor, performance loss is reduced from 0.6 dB to less than 0.1dB

FPGA implementation

FPGA Min-Sum Decoder

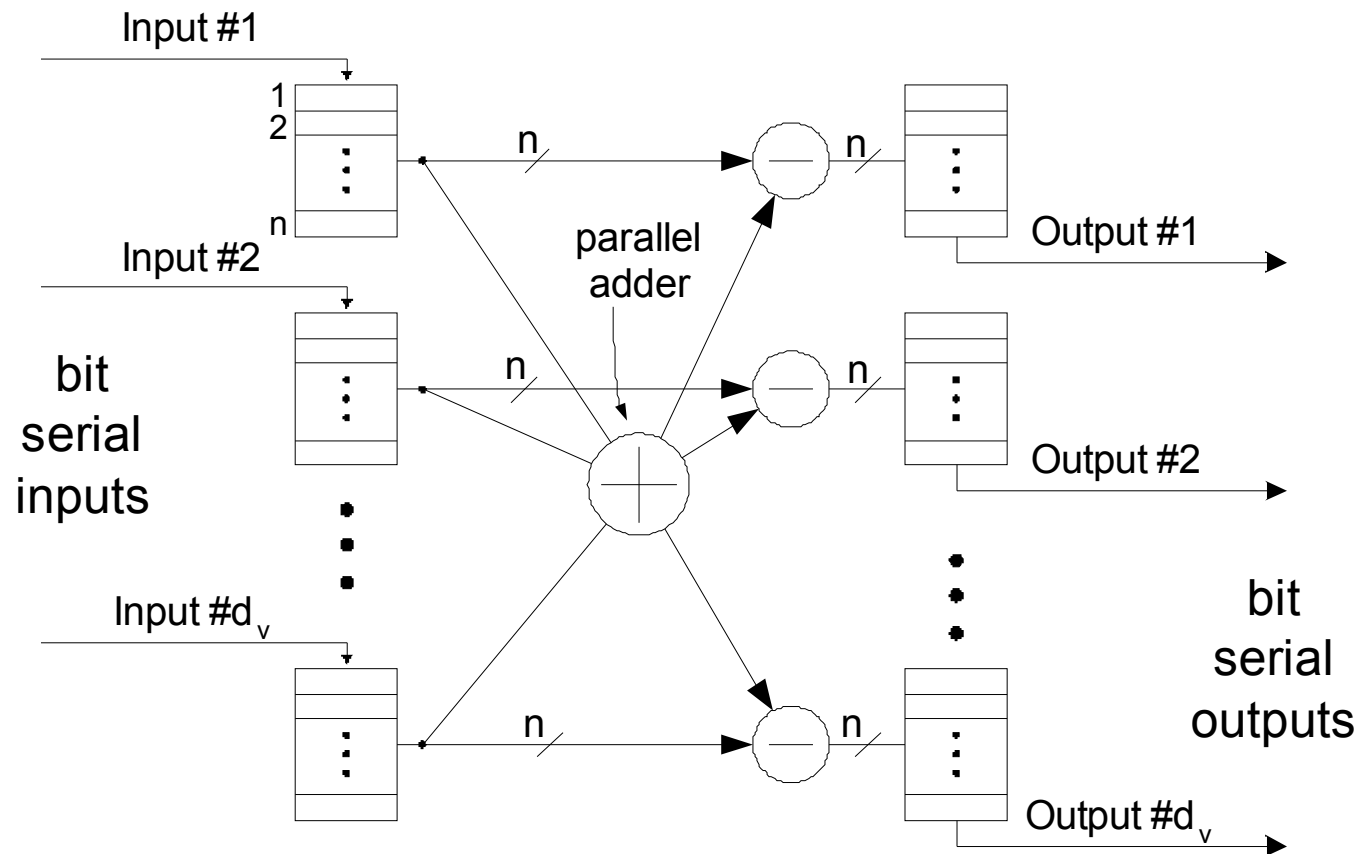
- Decoder built on FPGA (Stratix EP1S80)
- Fully parallel architecture
- Approximate Min-Sum decoding with correction
- Bit-serial message passing
- A regular-(4,15) (480, 355) LDPC code constructed using progressive edge growth (PEG) algorithm

Check Node Architecture



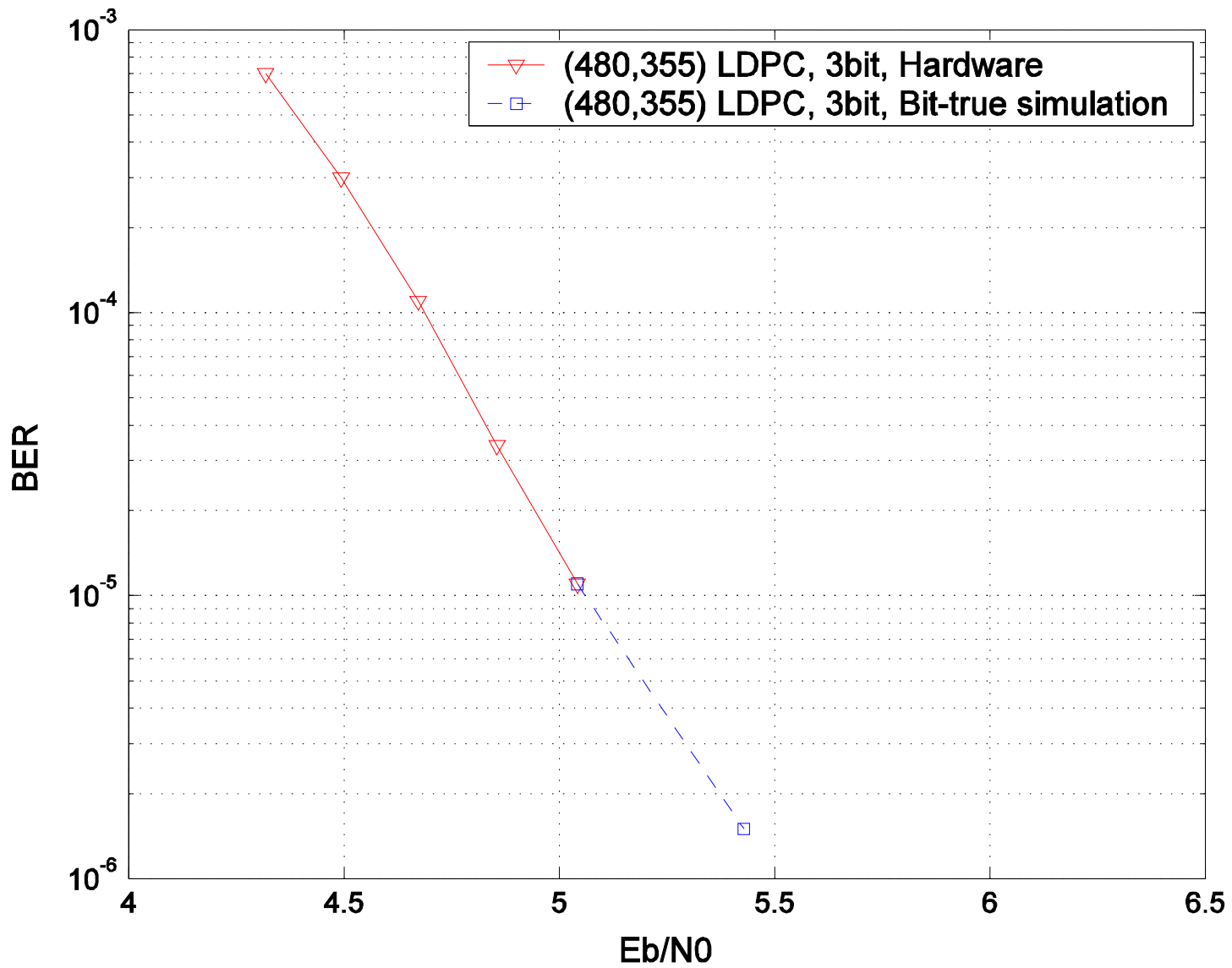
- Output sign is calculated separately using XOR gates

Variable Node Architecture



- Inputs arrive bit-serially, LSB first
- Addition and subtractions are performed bit-parallel
- For larger word lengths, a serial approach will be more efficient

BER Performance



Decoder Spec

	This work	L. Yang et al. TCAS 2006
FPGA device	Stratix EP1S80	Xilinx XC2V8000
Logic	66,000 LUTs	53,000 LUTs
Memory	0	102 blockRAMs
Code type	PEG_LDPC (4,15) regular	multi-rate codes
Code rate	0.74	$\frac{1}{2}$, $\frac{5}{8}$, $\frac{7}{8}$
Code length	480	9000
Decoding algorithm	Modified Min-Sum	Min-Sum
iterations/frame	15	24
Clock frequency	61 MHz	100 MHz
Throughput	650 Mbit/sec	Up to 40 Mbit/sec
Quantization	3 bit	-

Conclusion

- Bit-serial message passing
 - An efficient approach for implementing fully-parallel LDPC decoders
 - Reduces interconnection complexity
 - Suitable for Min-Sum decoding
- Modified Min-Sum decoding with the correction factor demonstrates high BER performance with 48% saving in check node logic