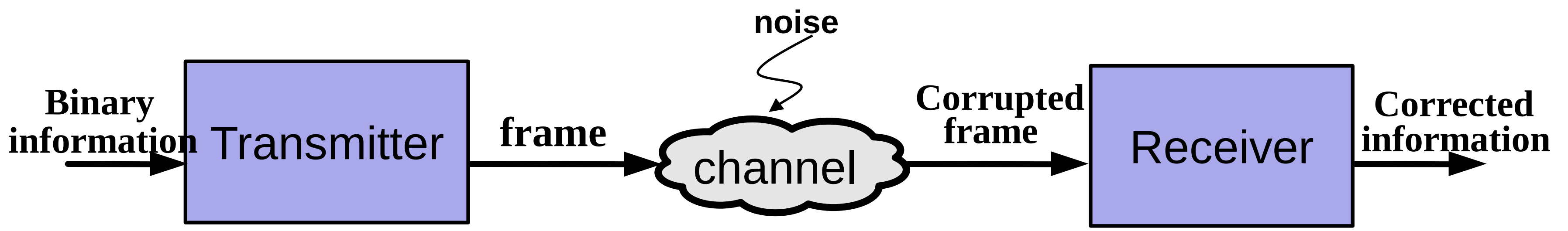
Error Correction and LDPC decoding

CMPE 691/491: DSP Hardware Implementation

Tinoosh Mohsenin

Error Correction in Communication Systems

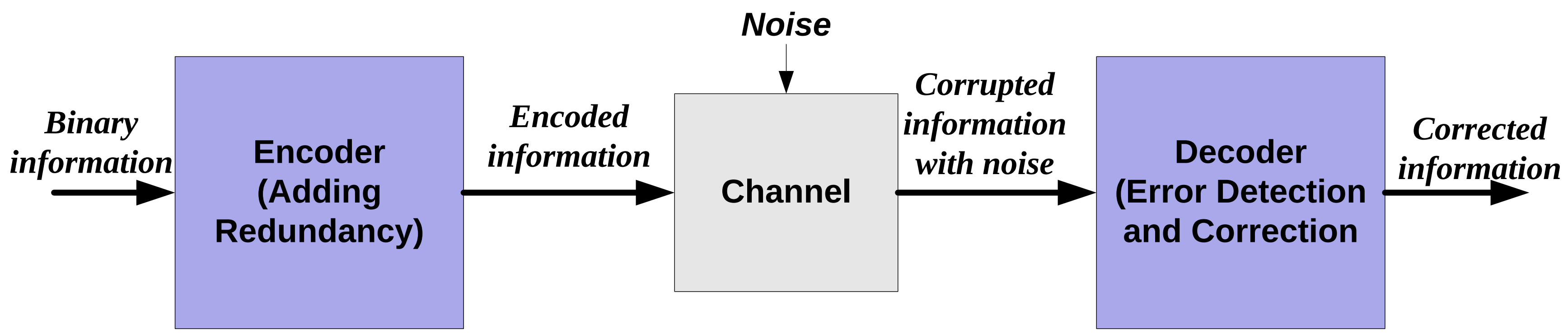


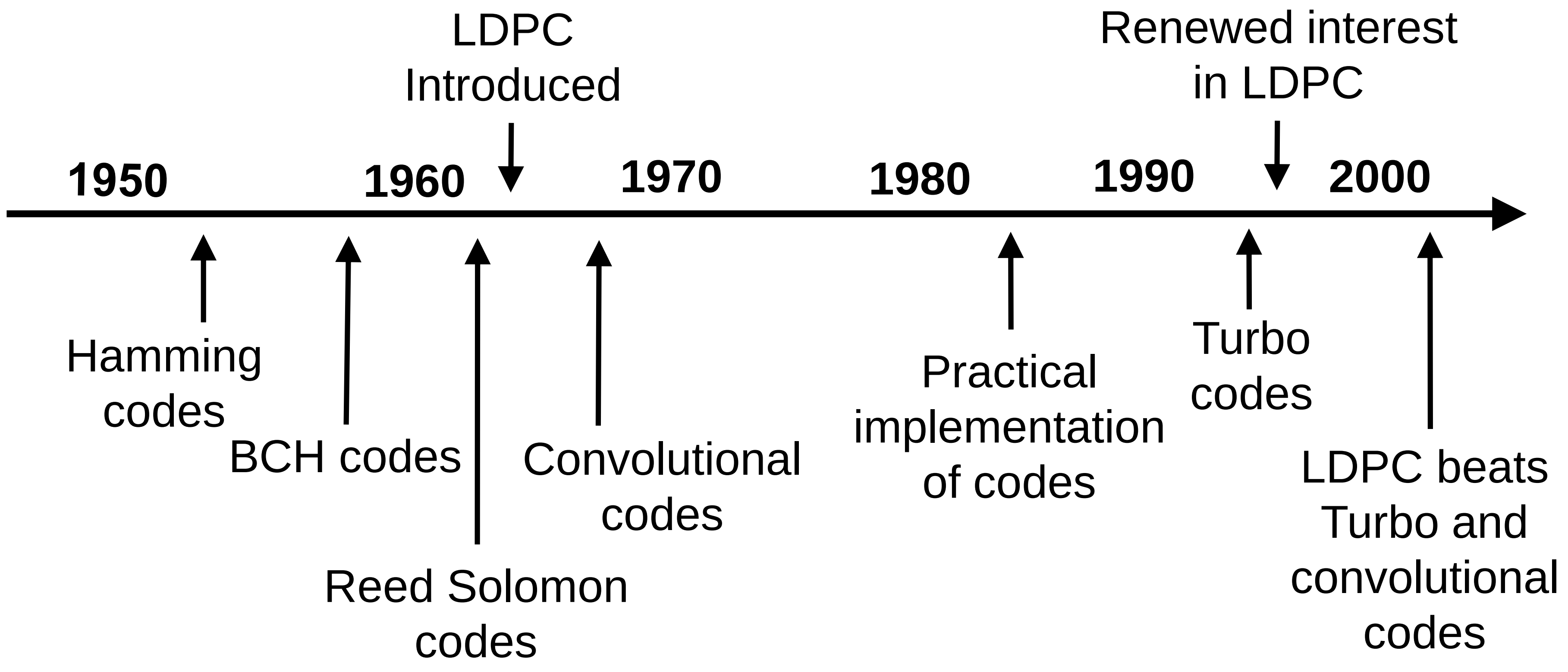
- Error: Given the original frame **k** and the received frame **k'**, how many corresponding bits differ?
  - Hamming distance (Hamming, 1950).
  - Example:
    - Transmitted frame: 1110011
    - Received frame: 1011001
    - Number of errors: 3

Error Detection and Correction

- Add extra information to the original data being transmitted.
  - Frame = **k** data bits + **m** bits for error control: n = k + m.
- Error detection: enough info to detect error.
  - Need retransmissions.
- Error correction: enough info to detect and correct error.
  - Forward error correction (FEC).

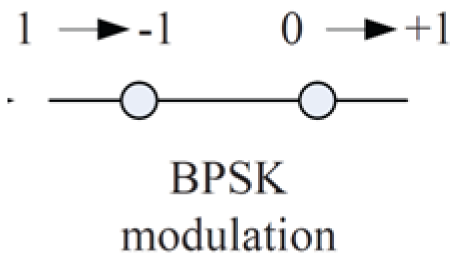
Error Correction in Communication Systems


Noise
Binary information
Encoder (Adding Redundancy)
Encoded information
Channel
Corrupted information with noise
Decoder (Error Detection and Correction
Corrected information

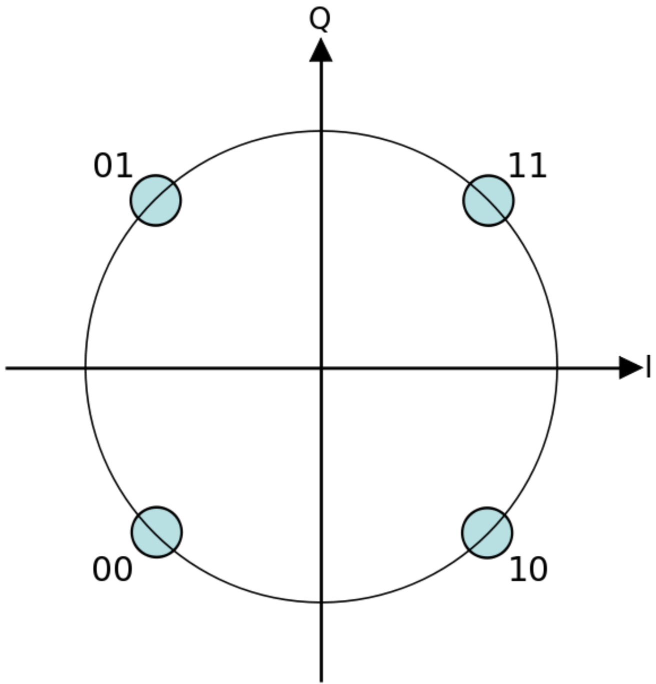


Modulation

- **Phase-shift keying (PSK)** is a digital modulation scheme that conveys data by changing, or modulating, the phase of a reference signal
- **Binary Phase Shift-Keying (BPSK) Modulation**
  - phase reversal keying, or 2PSK) is the simplest form of phase shift keying (PSK). It uses two phases which are separated by
  - In matlab: 1-2X where X is the input signal



- **Quadrature phase-shift keying (QPSK)**
  - 4-PSK, or 4-QAM. QPSK uses four points on the constellation diagram, equispaced around a circle. With four phases, QPSK can encode two bits per symbol, shown in the diagram with gray coding to minimize the bit error rate (BER).

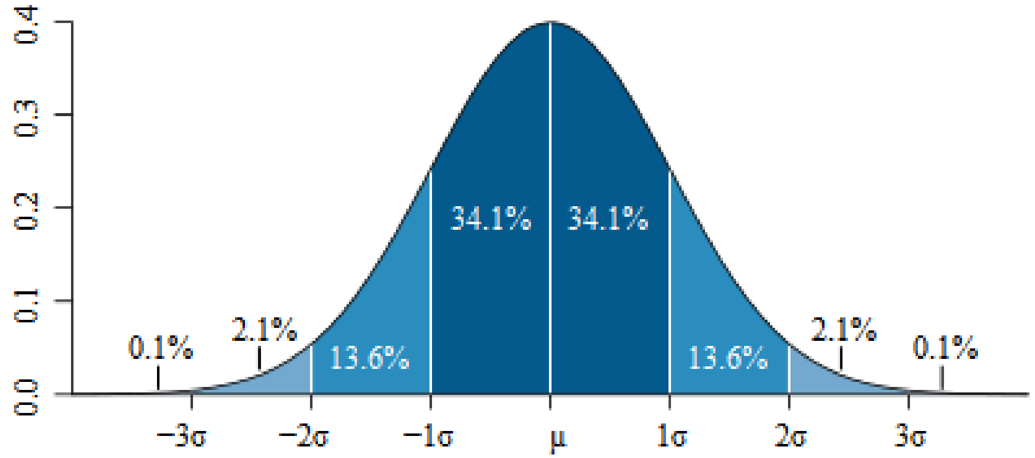


Key terms

- Encoder : adds redundant bits to the sender's bit stream to create a codeword.
- Decoder: uses the redundant bits to detect and/or correct as many bit errors as the particular error-control code will allow.
- Communication Channel: the part of the communication system that introduces errors.
  - Ex: radio, twisted wire pair, coaxial cable, fiber optic cable, magnetic tape, optical discs, or any other noisy medium
- Additive white Gaussian noise (AWGN)

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Larger noise makes the distribution wider



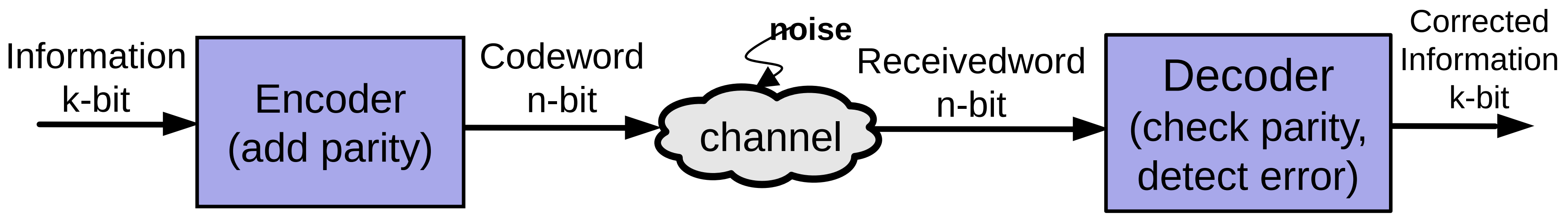
Important metrics

- Bit error rate (BER): The probability of bit error.
  - We want to keep this number small
  - Ex: BER=10^{-4} means if we have transmitted10,000 bits, there is 1 bit error.
  - BER is a useful indicator of system performance independent of error channel
  - BER=Number of error bits/ total number of transmitted bits
- Signal to noise ratio (SNR): quantifies how much a signal has been corrupted by noise.
  - defined as the ratio of signal power to the noise power corrupting the signal. A ratio higher than 1:1 indicates more signal than noise
  - often expressed using the logarithmic decibel scale:

$$\mathrm{SNR_{dB}} = 10\log_{10}\left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}}\right)$$

  - Important number: 3dB means signal power is two times noise power

Error Correction in Communication Systems



- **Goal: Attain lower BER at smaller SNR**
- Error correction is a key component in communication and storage applications.
- Coding example: Convolutional, Turbo, and Reed-Solomon codes
- What can 3 dB of coding gain buy?
  - A satellite can send data with half the required transmit power
  - A cellphone can operate reliably with half the required receive power

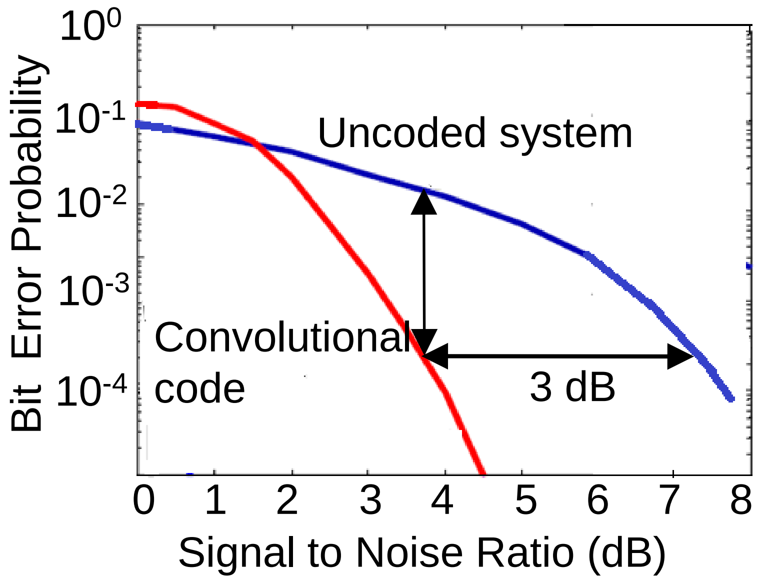


Figure courtesy of B. Nikolic, 2003 (modified)

LDPC Codes and Their Applications

- Low Density Parity Check (LDPC) codes have superior error performance
  - 4 dB coding gain over convolutional codes

- Standards and applications
  - 10 Gigabit Ethernet (10GBASE-T)
  - Digital Video Broadcasting (DVB-S2, DVB-T2, DVB-C2)
  - Next-Gen Wired Home Networking (G.hn)
  - WiMAX (802.16e)
  - WiFi (802.11n)
  - Hard disks
  - Deep-space satellite missions

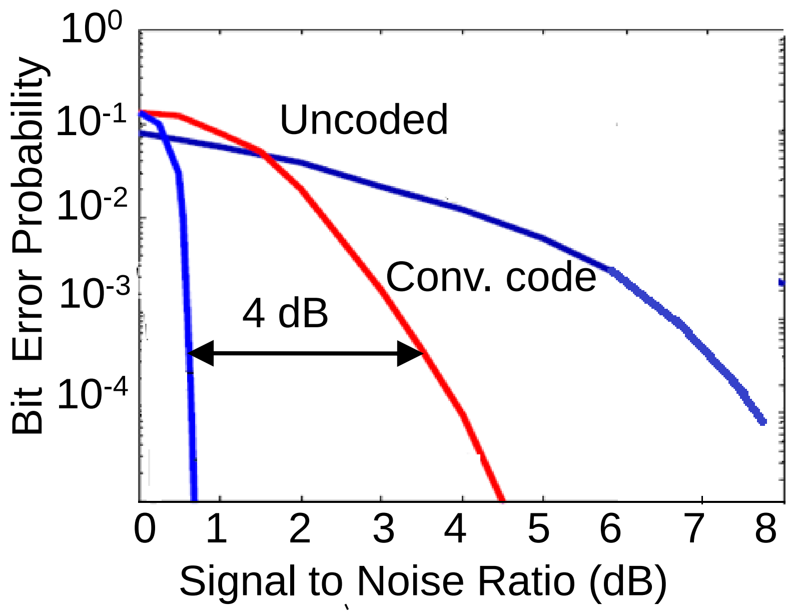
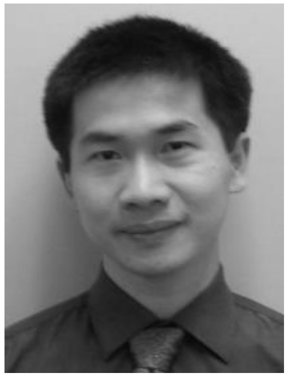
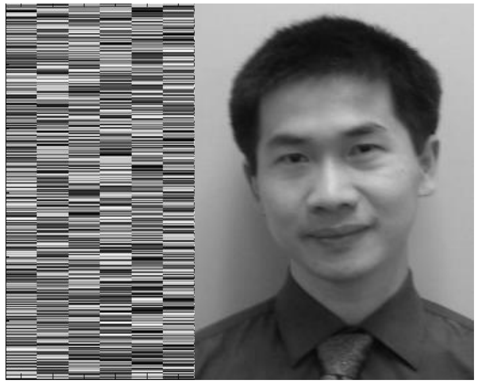


Figure courtesy of B. Nikolic, 2003 (modified)

Encoding Picture Example



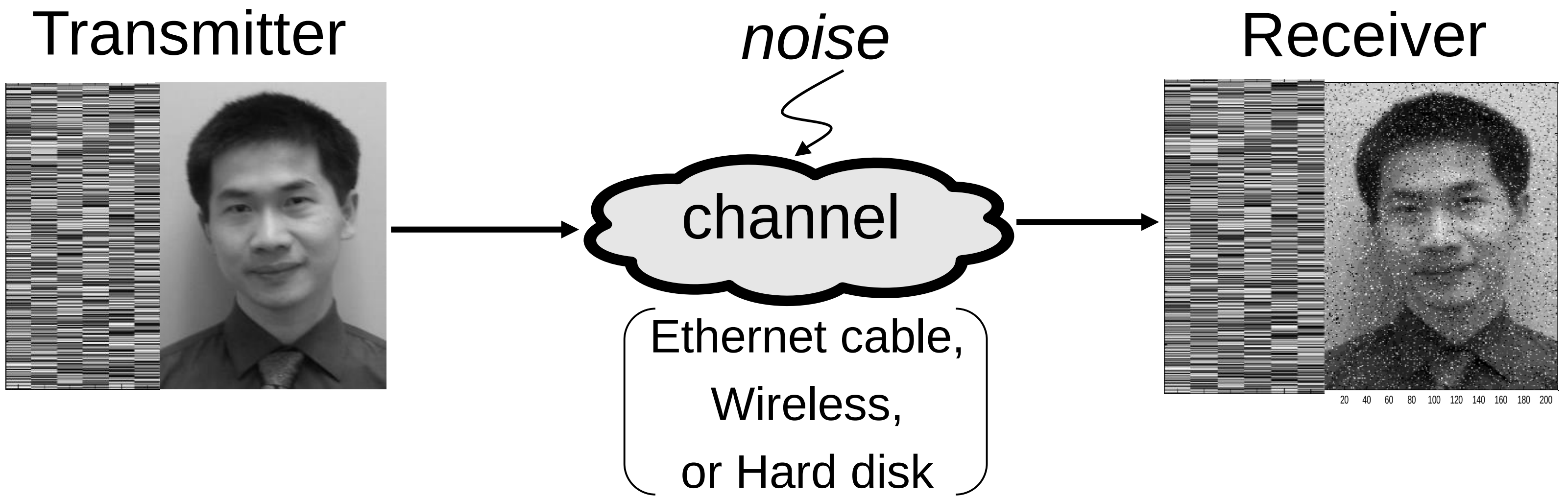


$$H=\begin{bmatrix} 1\,0\,0\,0\,0\,0\,0\,0\,0\,0\ldots 1\,1\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,1\,0 \\ 0\,1\,0\,0\,0\,0\,0\,0\,0\,0\ldots 1\,1\,1\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,1 \\ 0\,0\,1\,0\,0\,0\,0\,0\,0\,0\ldots 0\,0\,1\,1\,0\,1\,1\,0\,0\,0\,1\,1\,0\,0\,0\,1\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\ldots 0\,0\,0\,1\,1\,0\,1\,1\,0\,0\,0\,1\,1\,0\,0\,0\,1 \\ \ldots \qquad\qquad\qquad \ldots \end{bmatrix} \quad V=\begin{bmatrix} 1\,0\,1\,1\,1\,0\,1\,1\,0\,0\,0\,0 \ldots 1\,1\,1\,1\,0\,0\,0\,1\,1\,1\,1\,1 \end{bmatrix}$$

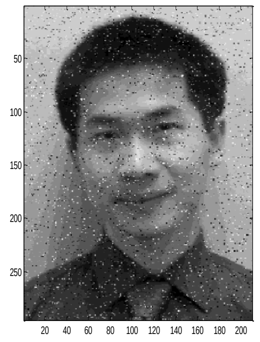
$$H.V_i^T=0$$

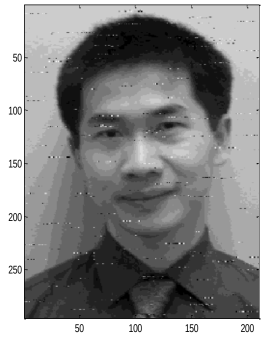
Binary multiplication called syndrome check

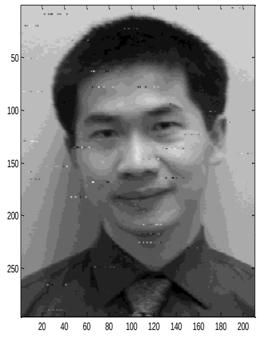
Decoding Picture Example

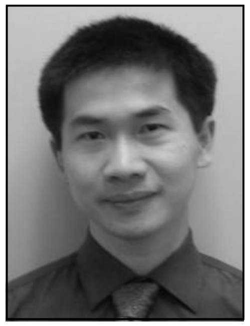


Iterative message passing decoding


Iteration 1


Iteration 5


Iteration 15


Iteration 16

LDPC Codes—Parity Check Matrix

- Defined by a large binary matrix, called a *parity check matrix* or *H* matrix
  - Each row is defined by a parity equation
  - The number of columns is the code length

- Example: 6x 12 *H* matrix for a12-bit LDPC code
  - No. of columns=12 (i.e. Receivedword (*V*) = 12 bit)
  - No. of rows= 6
  - No. of ones per row=3 (row weight)
  - No. of ones per col= 2 (column weight)

$$H = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

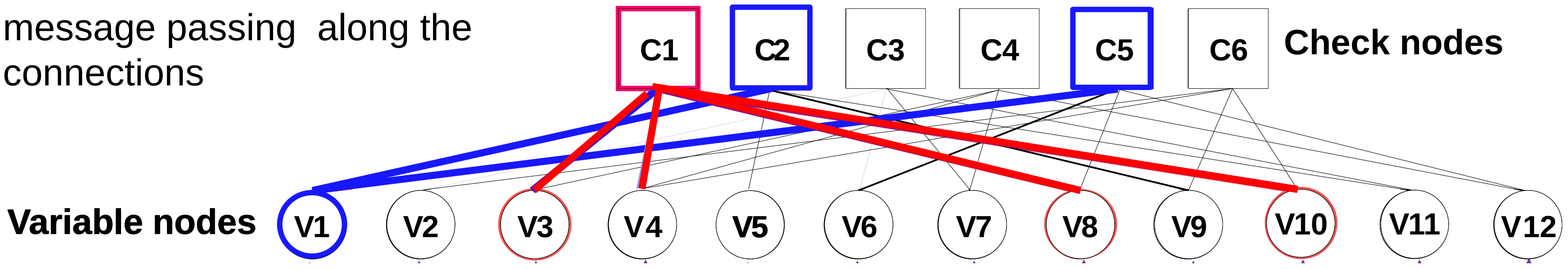
LDPC Codes—Tanner Graph

- Interconnect representation of *H* matrix
  - Two sets of nodes: Check nodes and Variable nodes
  - Each row of the matrix is represented by a Check node
  - Each column of matrix is represented by a Variable node
- A message passing method is used between nodes to correct errors

(1) Initialization with *Receivedword*
(2) Messages passing until correct
Example:

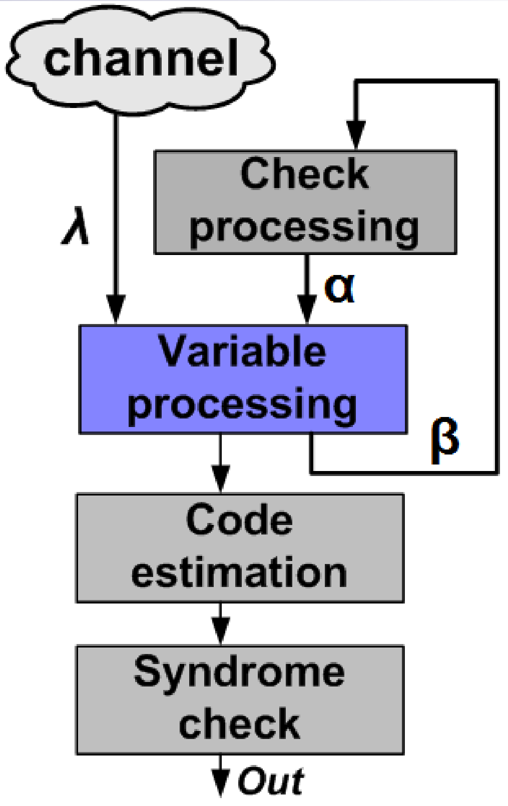
$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

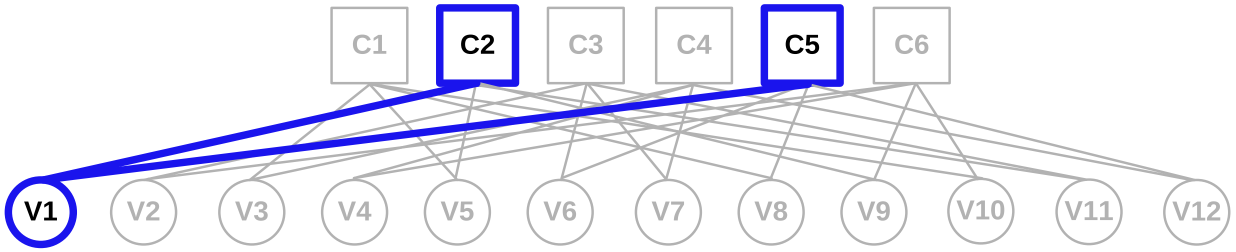
The same for other nodes: message passing along the connections



Message Passing: Variable node processing

$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$





$$Z_j = \sum_{j', h_{ij'}=1} \alpha_{ij'} + \lambda_j$$

$$\beta_{ij} = Z_j - \alpha_{ij}$$

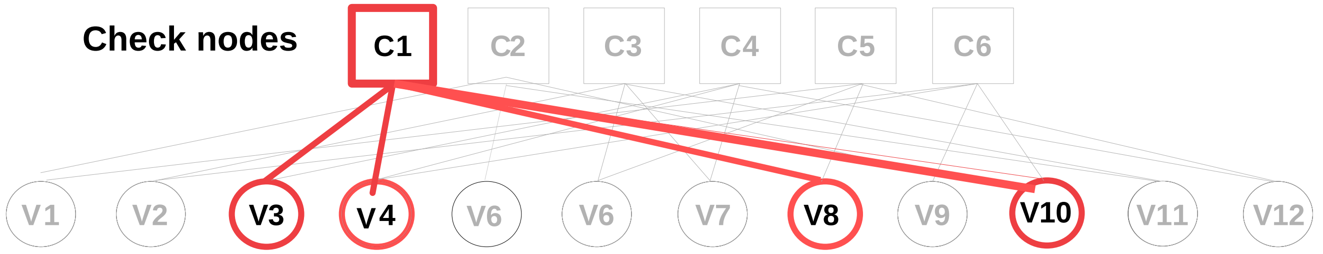
α: message from check to variable node

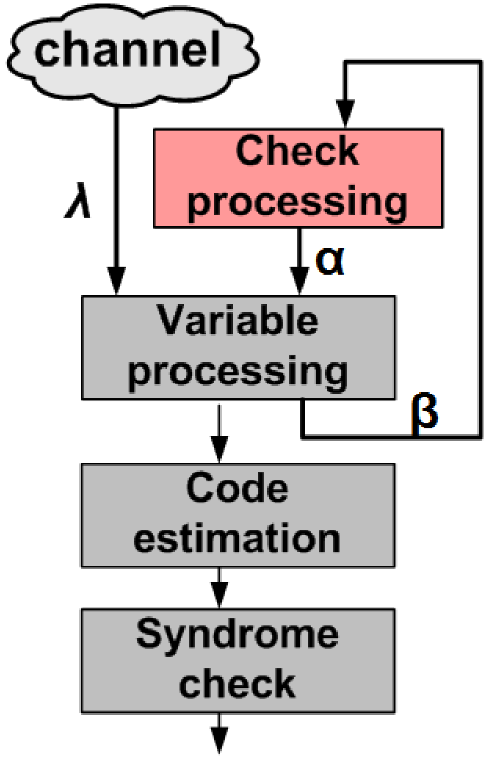
β: message from variable to check node

λ is the original received information from the channel

Message Passing: Check node processing (MinSum)

$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$





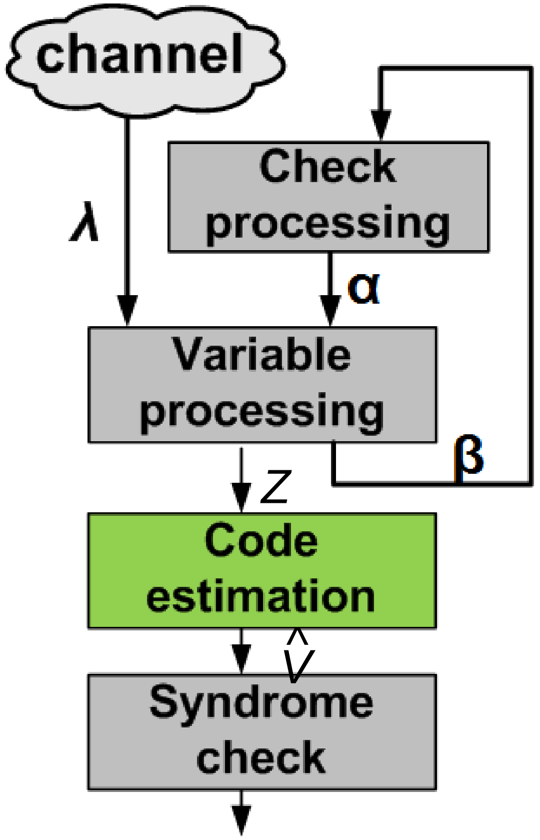
$$\alpha_{ij\,MS} = \left(\underbrace{\left(\prod_{j',h_{ij'}=1,\,j'\neq j} sign\beta_{ij'} \right)}_{\textbf{Sign}} \times \underbrace{\min_{j',h_{ij'}=1,\,j'\neq j} \left(\left|\beta_{ij'}\right| \right)}_{\textbf{Magnitude}} \right) \times Sfactor$$

After check node processing, the next iteration starts with another variable node processing (begins a new iteration)

Code Estimation

- Based on your modulation scheme (here BPSK) estimate the transmitted bits

$$\hat{V} = \begin{cases} 1, & \text{if } z_i \leq 0 \\ 0, & \text{if } z_i > 0 \end{cases}$$

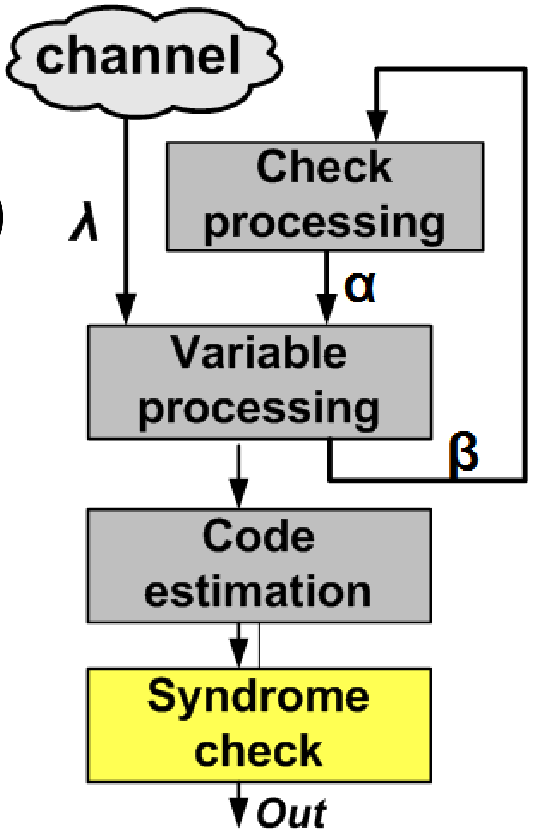


Syndrome Check

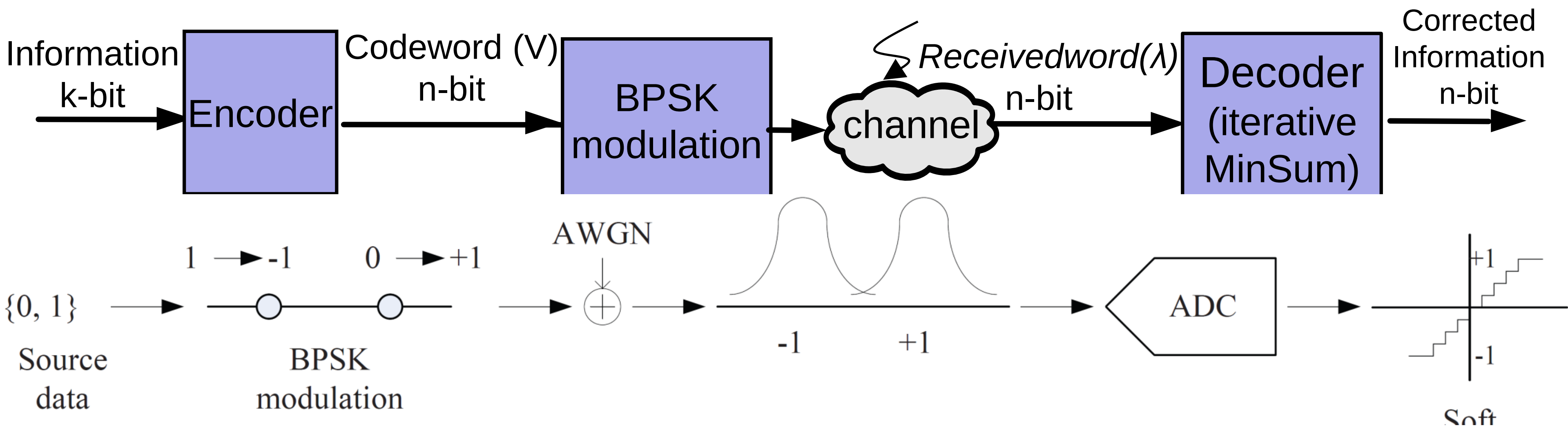
- Compute syndrome

$$H.\hat{V}_i^T=0$$ (Binary multiplication)

- Ex:
  - If syndrome =0, terminate decoding
  - Else, continue another iteration



Example



Encoded information V= [1 0 1 0 1 0 1 0 1 1 1 1]

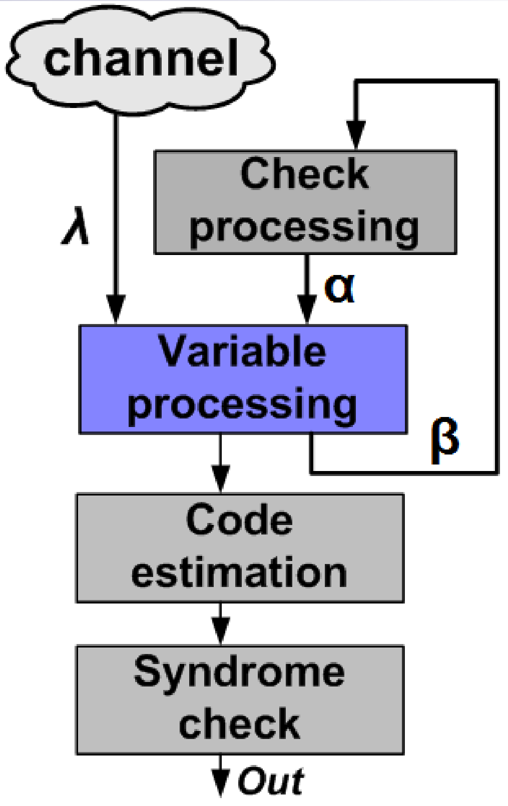
BPSK modulated= [-1 1 -1 1 -1 1 -1 1 -1 -1 -1 -1]

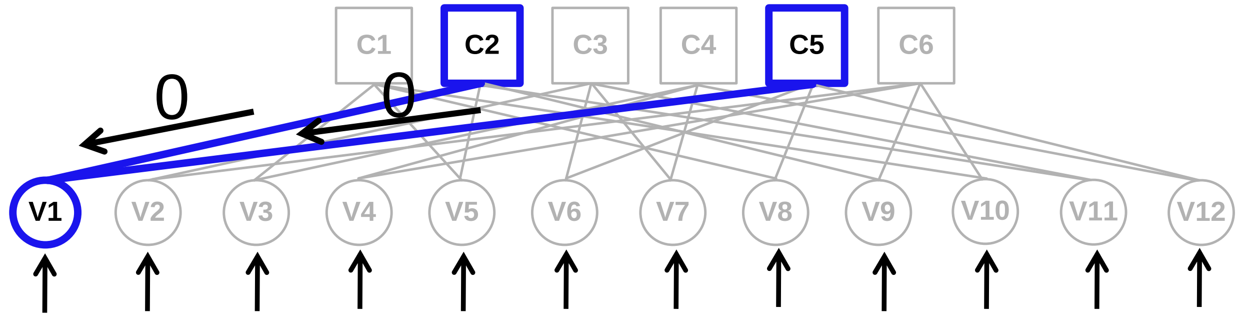
λ (Received data from channel)=

[-9.1 4.9 -3.2 3.6 -1.4 3.1 0.3 1.6 -6.1 -2.5 -7.8 -6.8]

Ex: Variable node processing (iteration 1)

$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$





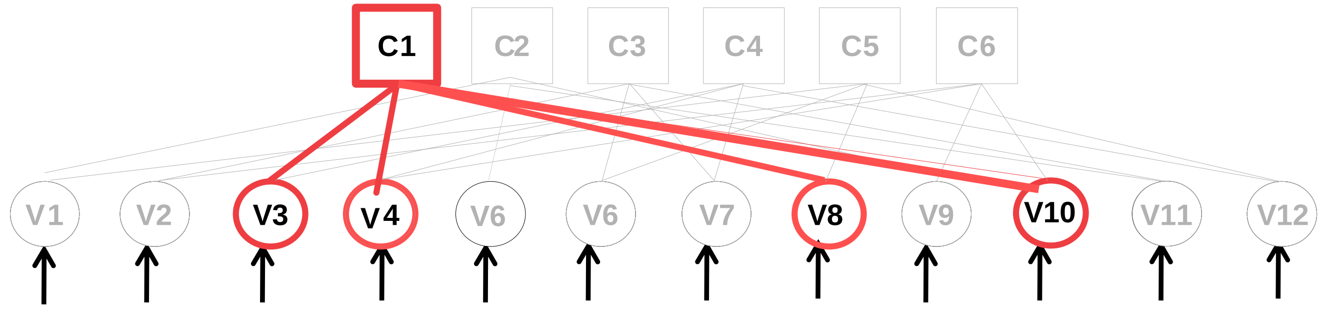
λ= -9.1 4.9 -3.2 3.6 -1.4 3.1 0.3 1.6 -6.1 -2.5 -7.8 -6.8

$$Z_1 = \sum_{j',h_{ij'}=1} \alpha_{i1} + \lambda_1$$

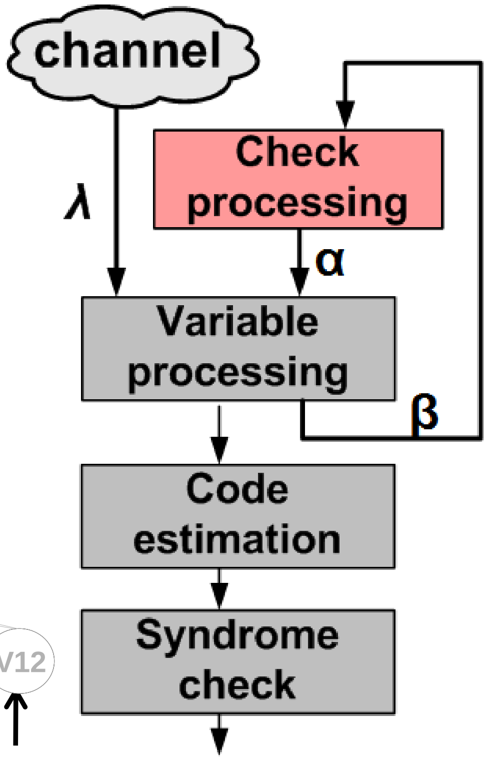
$$\beta_{12} = \qquad\qquad \beta_{15} =$$

Ex: Check node processing (Iteration 1)

$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$



β =-9.1 4.9 -3.2 3.6 -1.4 3.1 0.3 1.6 -6.1 -2.5 -7.8 -6.8



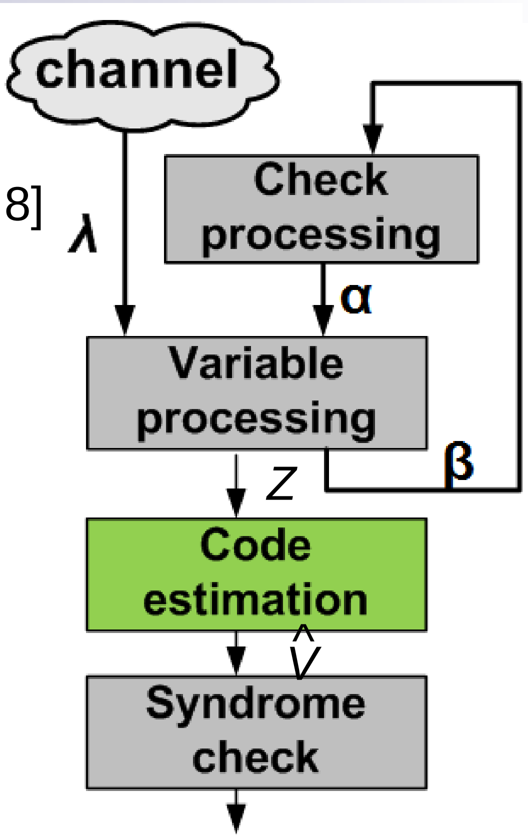
$|\alpha_{13}| = Min\{3.6, 1.6, 2.5\}$ $|\alpha_{14}| =$ $|\alpha_{18}| =$ $|\alpha_{110}| =$

$Sign(\alpha_{13}) = (1)(1)(-1) = -1$ $Sign(\alpha_{14}) =$ $Sign(\alpha_{18}) =$ $Sign(\alpha_{110}) =$

- Here assume $Sfactor = 1$

Ex: Code Estimation (Iteration 1)

$Z = \lambda$ =
[-9.1 4.9 -3.2 3.6 -1.4 3.1 0.3 1.6 -6.1 -2.5 -7.8 -6.8]



$$\hat{V} = \begin{cases} 1, & \text{if } z_i \le 0 \\ 0, & \text{if } z_i > 0 \end{cases}$$

$\hat{V}$ = 1 0 1 0 1 0 0 0 1 1 1 1

Ex: Syndrome Check (iteration 1)

- Compute syndrome

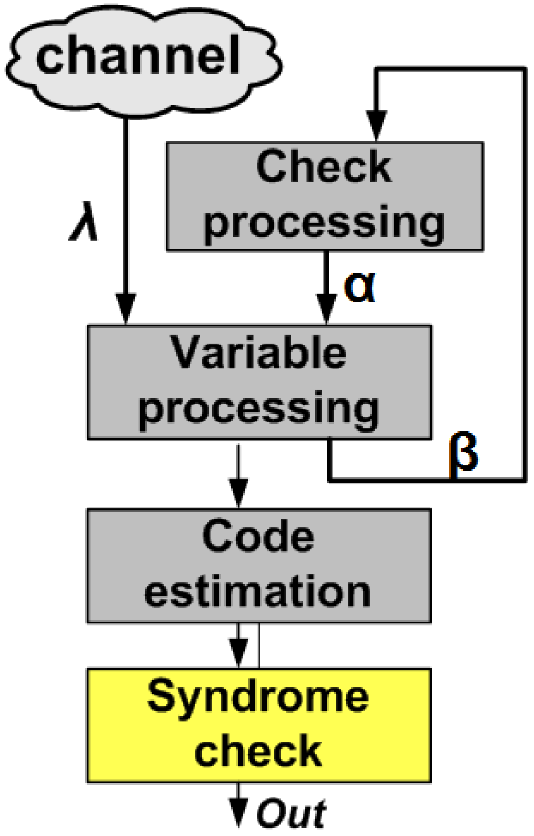
$H.\hat{V}_i^T = 0$ (Binary multiplication)

$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$Syndrome_i = \underset{j \in h_{ij} | h_{ij} \neq 0}{XOR}(\hat{v}_j)$$

$$Sum\,Syndrom = \sum_i Syndrome_i$$

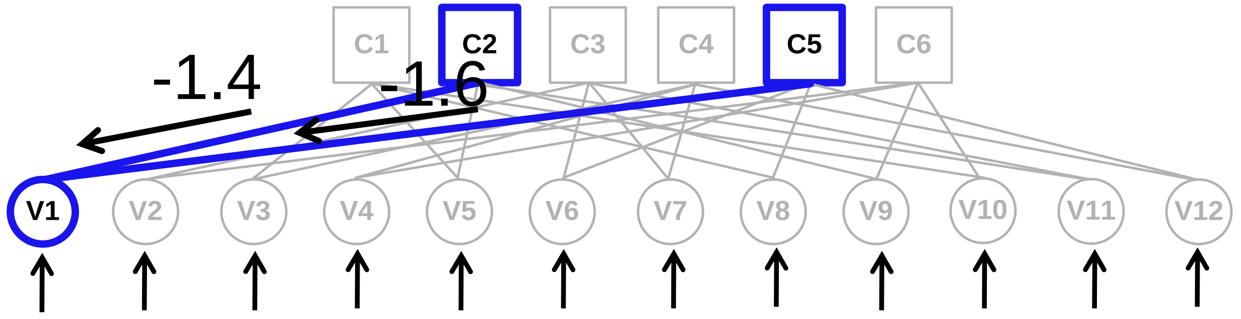
Sumsyndrome=2 Not ZERO => Error, continue decoding



Second iteration

- In variable node processing, compute β, α and Z based on the algorithm

$$H = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$



λ= -9.1 4.9 -3.2 3.6 -1.4 3.1 0.3 1.6 -6.1 -2.5 -7.8 -6.8

Z= [-12.1 7.1 -4.5 7.7 -7.2 4.4 -4.2 7.2 -10.0 -7.7 -8.9 -8.1]

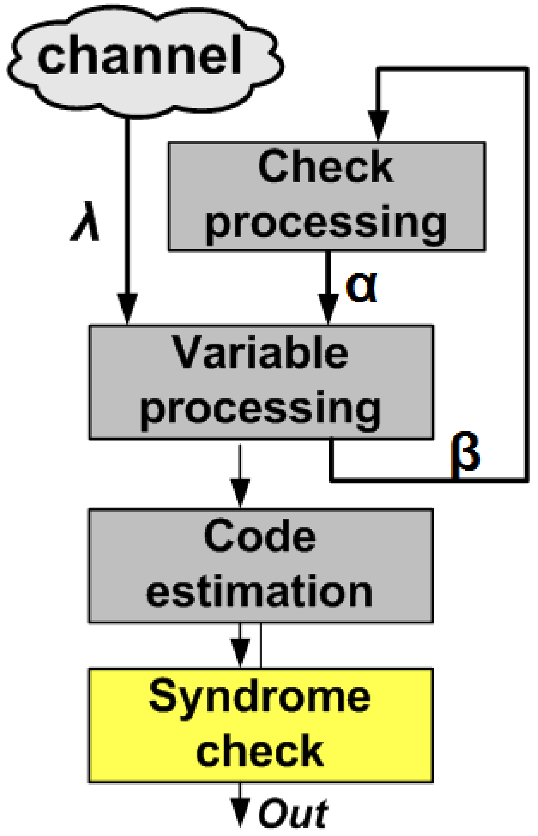
$\hat{V}$= [1 0 1 0 1 0 1 0 1 1 1 1]

Ex: Syndrome Check (iteration 2)

- Compute syndrome

$H.\hat{V}_i^T=0$ (Binary multiplication)

$$H=\begin{bmatrix} 0&0&1&1&0&0&0&1&0&1&0&0\\ 1&0&0&0&1&0&0&0&1&0&1&0\\ 0&1&0&0&0&1&1&0&0&0&1&0\\ 0&0&1&0&1&0&1&0&0&0&0&1\\ 1&0&0&0&0&1&0&1&0&0&0&1\\ 0&1&0&1&0&0&0&0&1&1&0&0 \end{bmatrix} \times \begin{bmatrix}1\\0\\1\\0\\1\\0\\1\\0\\1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\end{bmatrix}$$

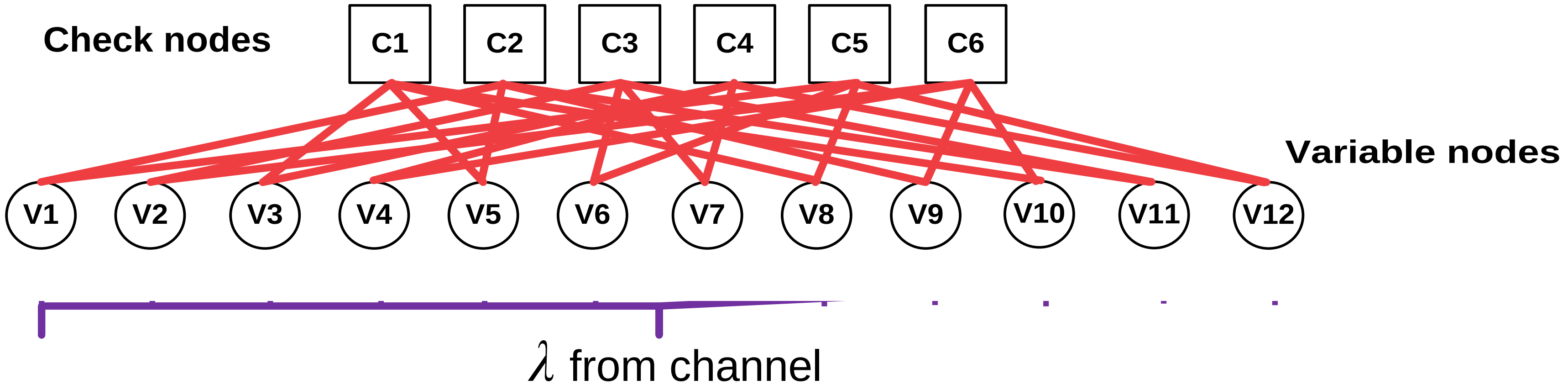


$$Syndrome_i = \underset{j\in h_{ij}|h_{ij}\neq 0}{XOR}(\hat{v}_j)$$

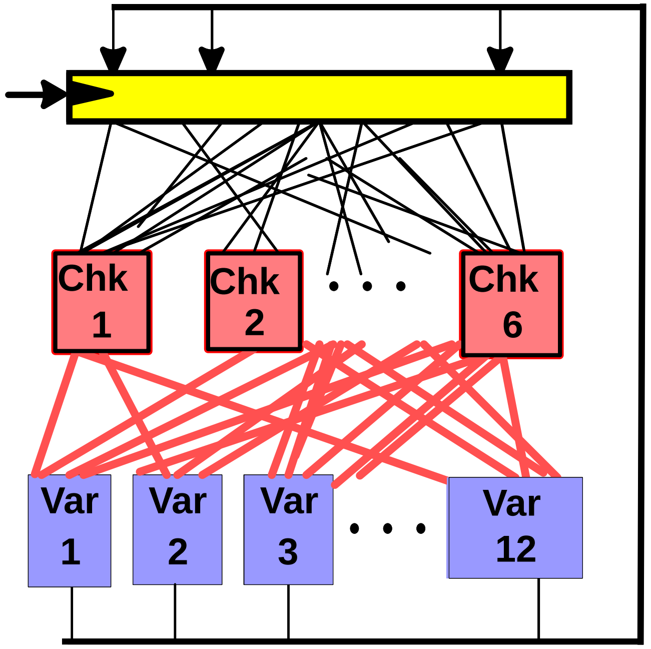
$$Sum\,Syndrom = \sum_i Syndrome_i$$

Sumsyndrome= ZERO => corrected code Terminate Decoding

Full-Parallel Decoding

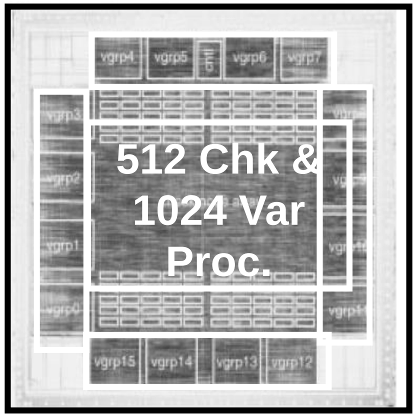


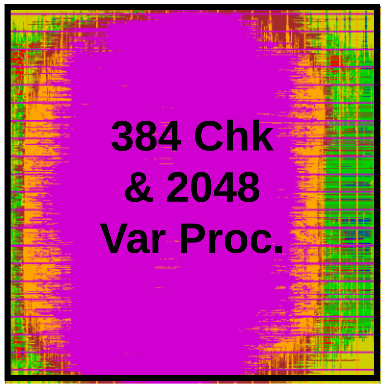
- Every check node and variable node is mapped to a processor
- All processors directly connected based on the Tanner graph
  - Very High throughput
  - No large memory storage elements (e.g. SRAMs)
  - High routing congestion
  - Large delay, area, and power caused by long global wires

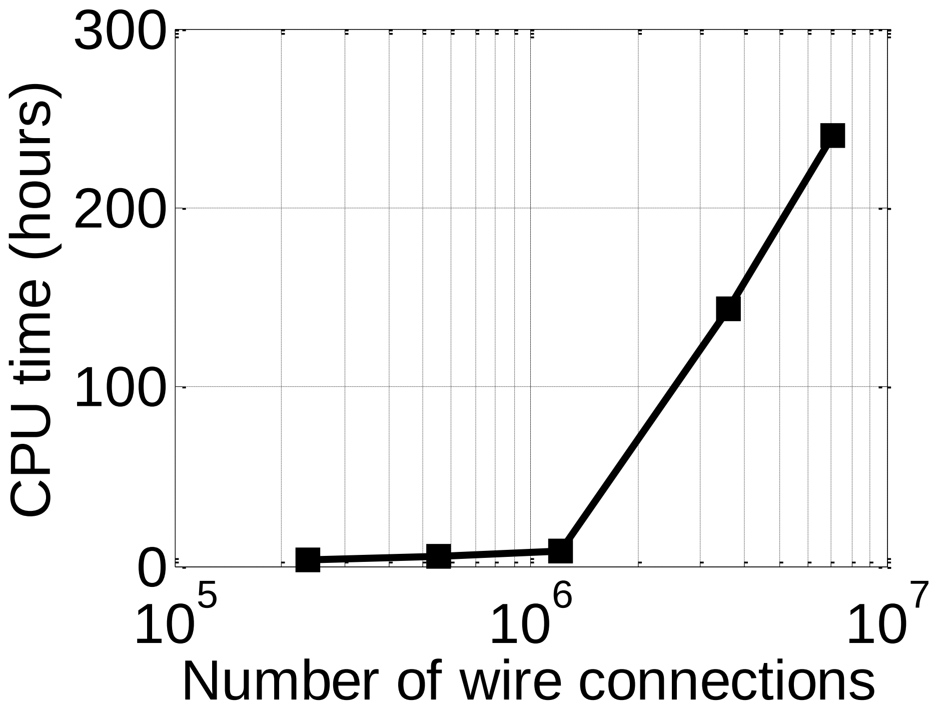


Full-Parallel LDPC Decoder Examples

- Ex 1: 1024-bit decoder, [*JSSC* 2002]
  - 52.5 mm^2, 50% logic utilization, 160 nm CMOS
- Ex 2: 2048 bit decoder, [*ISCAS* 2009]
  - 18.2 mm^2, 25% logic utilization, 30 MHz, 65 nm CMOS
  - CPU time for place & route>10 days



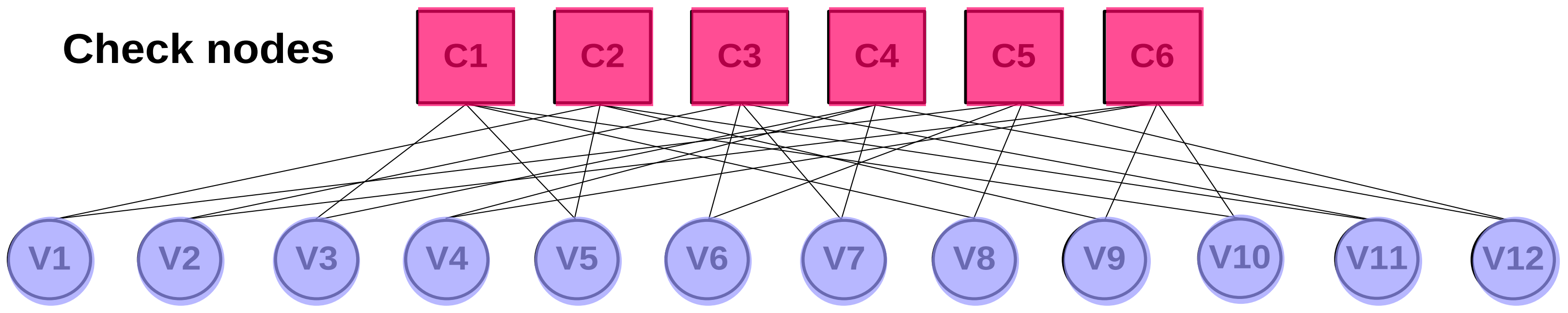




For all data in the plot:

- Same automatic place & route flow is used
- CPU: Quad Core, Intel Xeon 3.0GHz

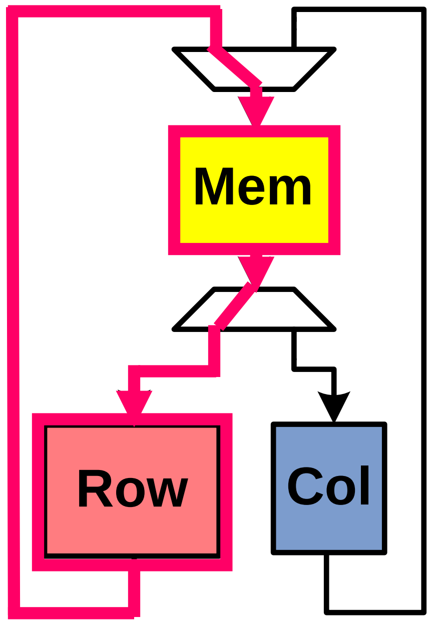
Serial Decoder Example



Check nodes

Variable nodes

(1) *initialize memory (clear contents)*



(2) *compute* *V1 V2 V3*

and store *V4 V5 V6*

V7 V8 V9

V10 V11 V12

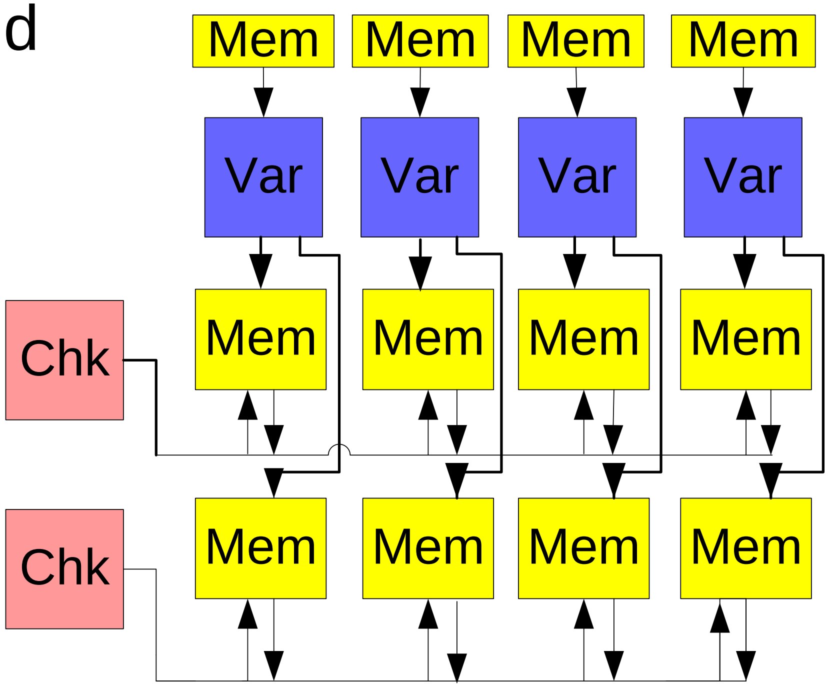
(3) *…now compute* *C1 C2 C3*

and store *C4 C5 C6*

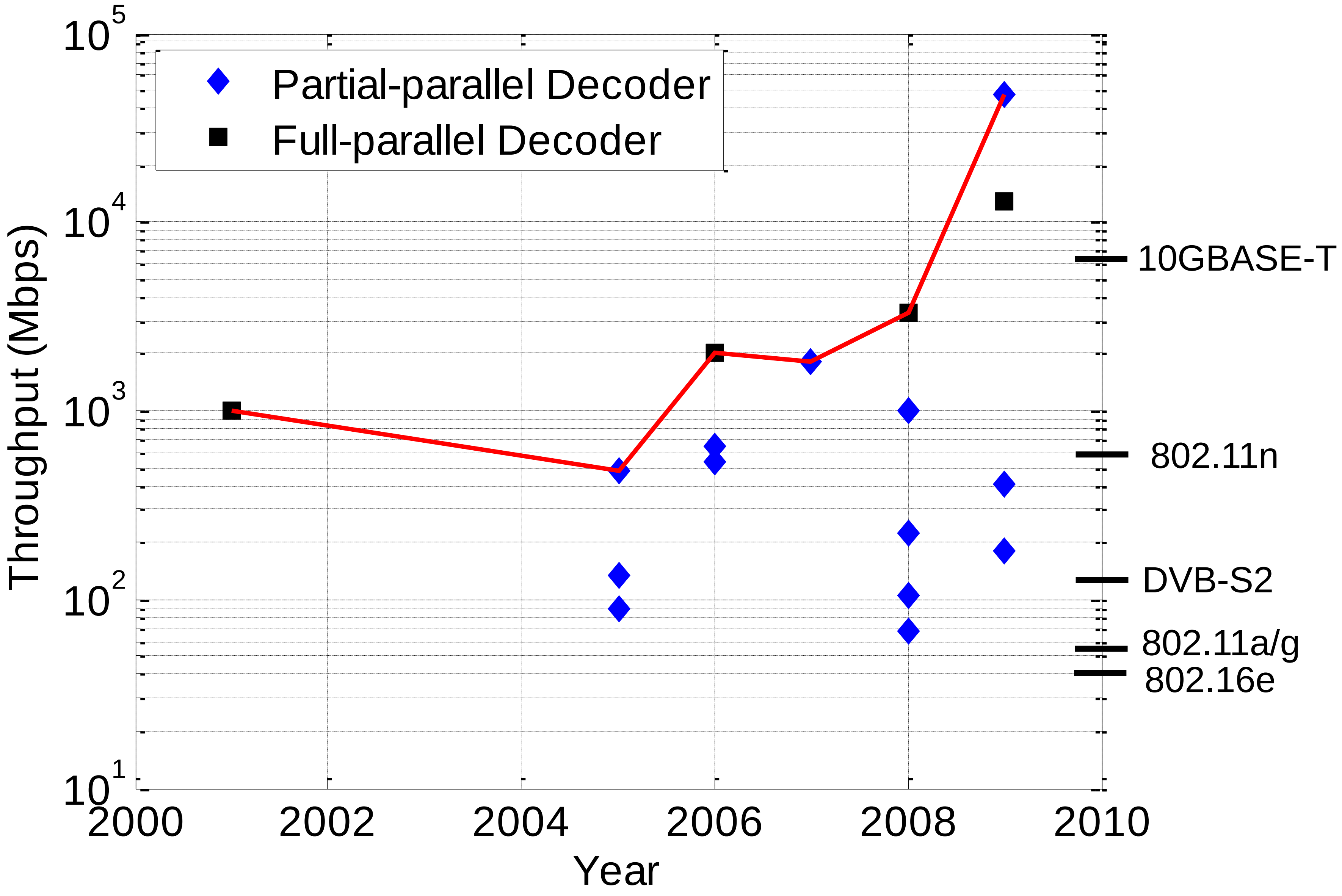
Decoding Architectures

- Partial parallel decoders
  - Multiple processing units and shared memories
  - Throughput: 100 Mbps-Gbps
  - Requires Large memory (depending on the size)
  - Requires Efficient Control and scheduling

$$H = \left[\begin{array}{ccc|ccc|ccc|ccc}
0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
\hline
0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{array}\right]$$

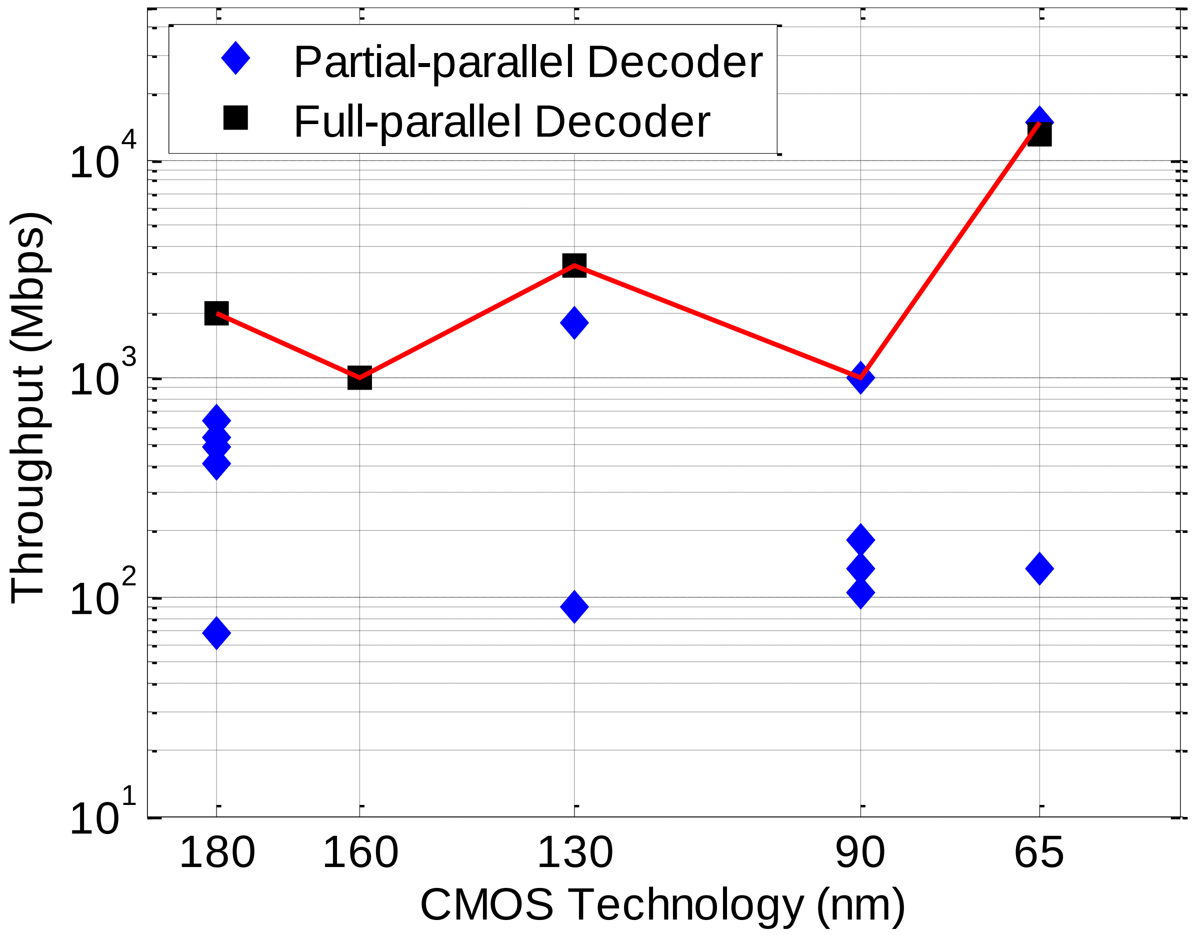


Reported LDPC Decoder ASICs



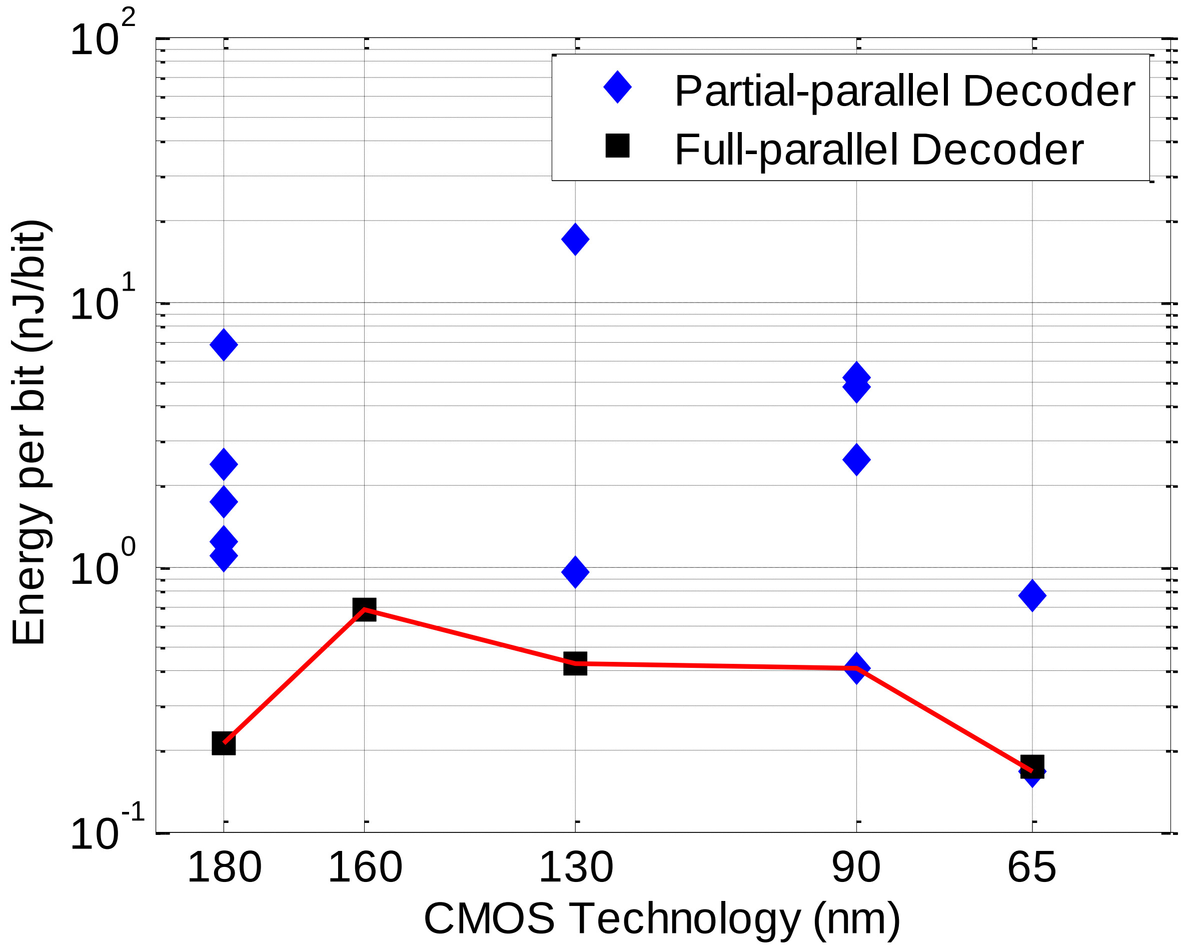
Throughput Across Fabrication Technologies

- Existing ASIC implementations without early termination
- Full-parallel decoders have the highest throughput



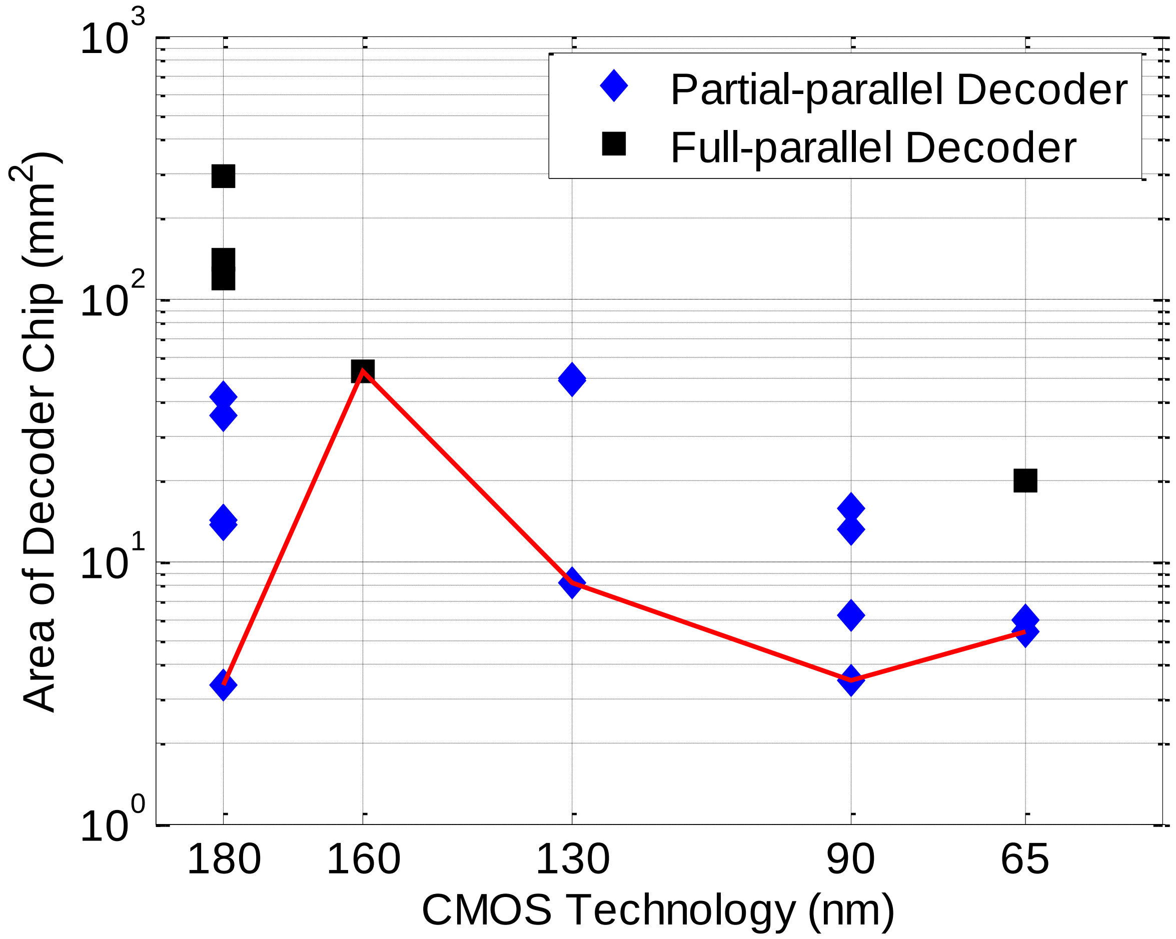
Energy per Decoded Bit in Different Technologies

- Existing ASIC implementations without early termination
- Full-parallel decoders have the lowest energy dissipation



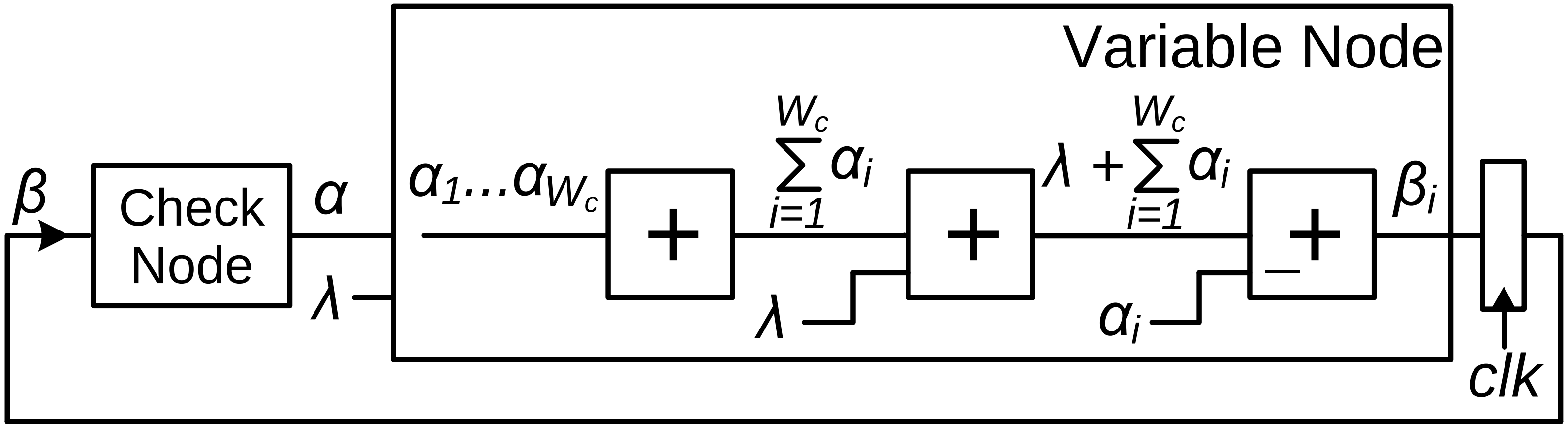
Circuit Area in Different Technologies

- Full-parallel decoders have the largest area due to the high routing congestion and low logic utilization



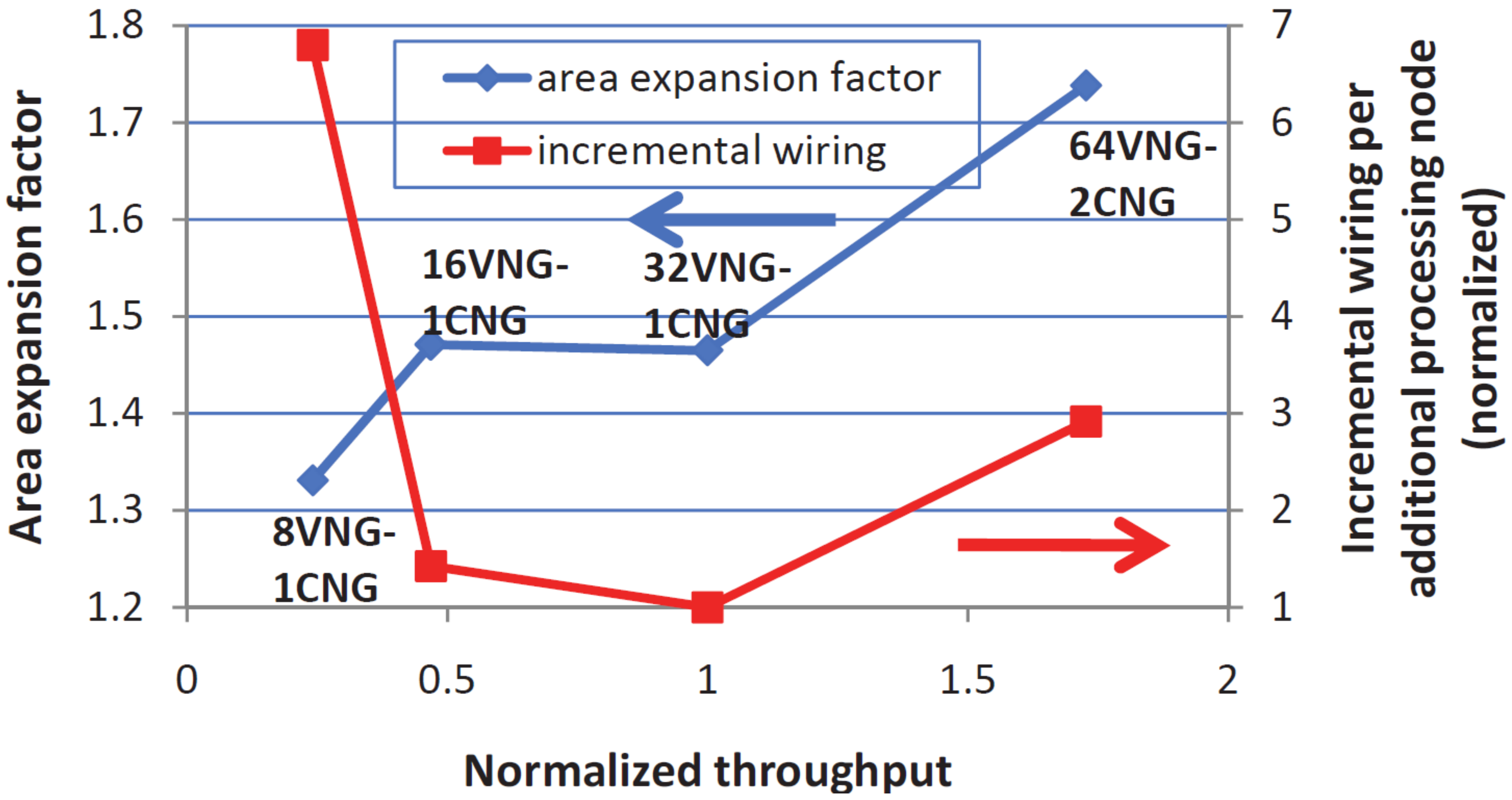
Key optimization factors

- Architectural optimization
  - Parallelism
  - Memory
- Data path wordwidth (fixedpoint format)

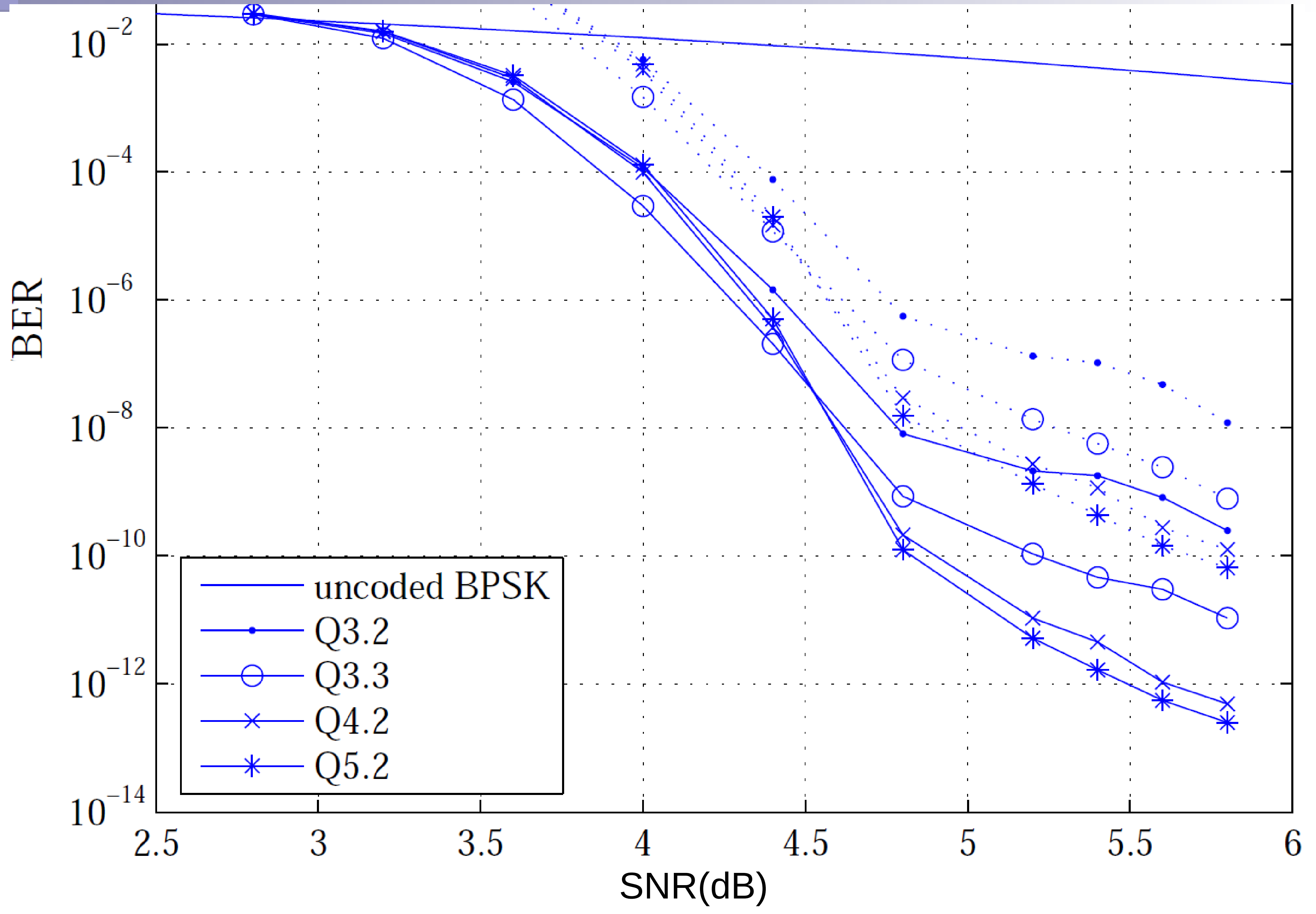


Variable Node
Check Node
β
α
λ
$\alpha_1 \dots \alpha_{W_c}$
$\sum_{i=1}^{W_c} \alpha_i$
$\lambda + \sum_{i=1}^{W_c} \alpha_i$
λ
α_i
β_i
+
+
+
_
clk

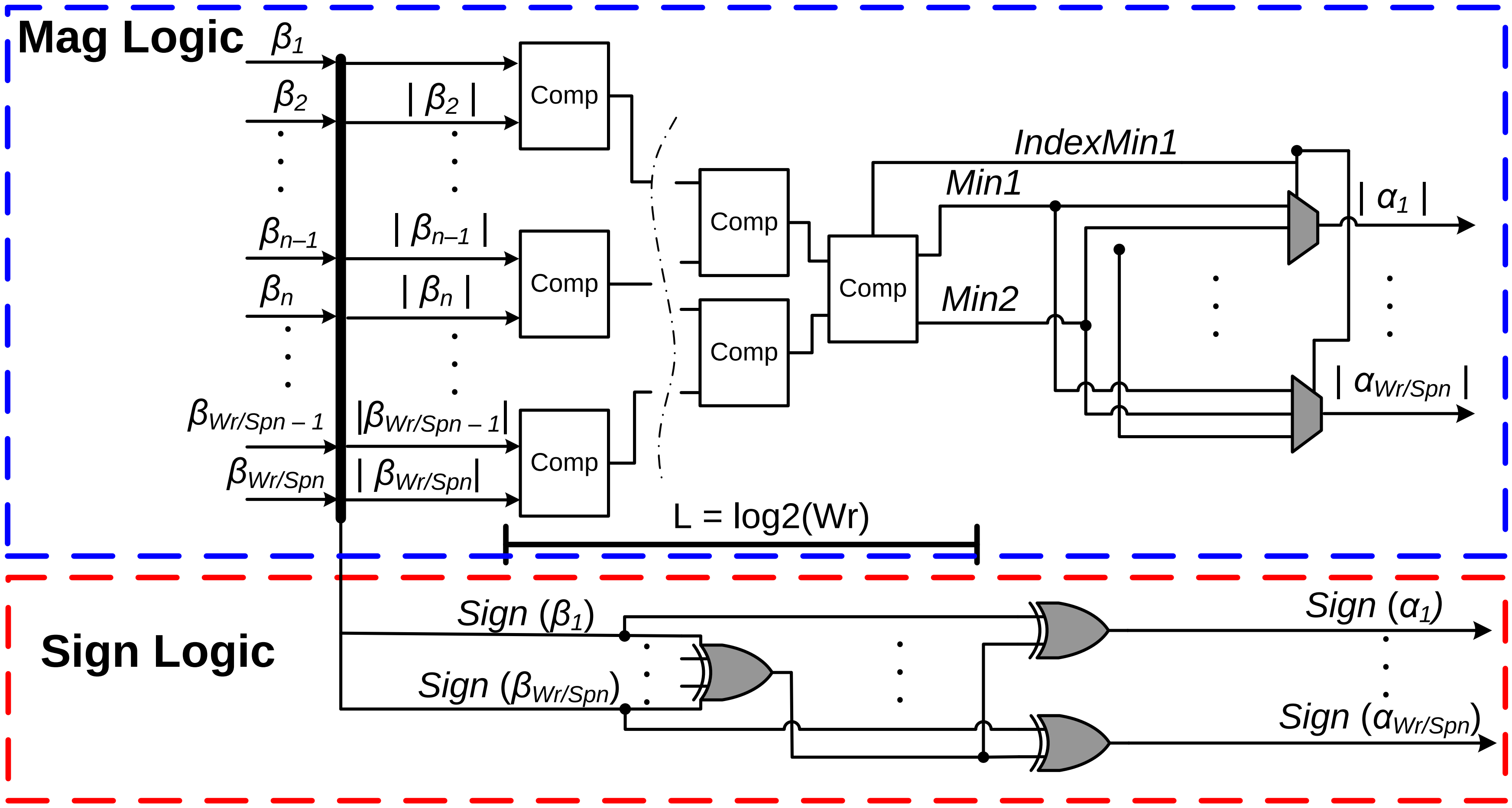
Architectural optimization

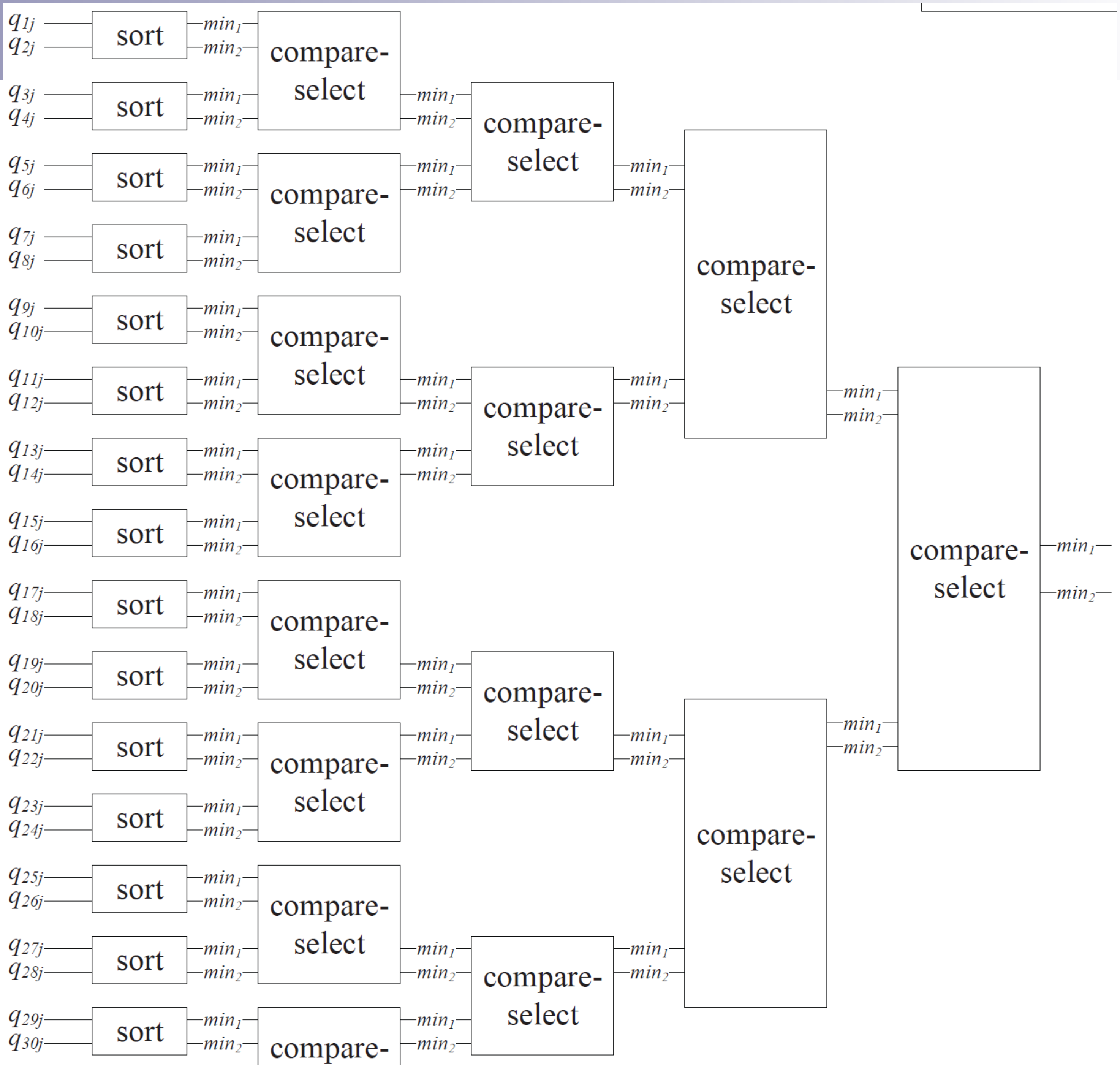


BER performance versus quantization format



Check Node Processor

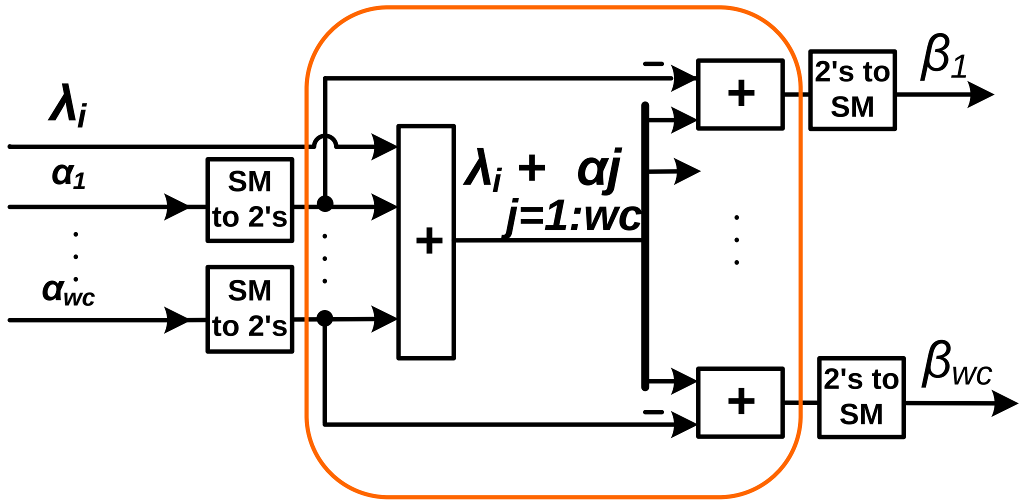


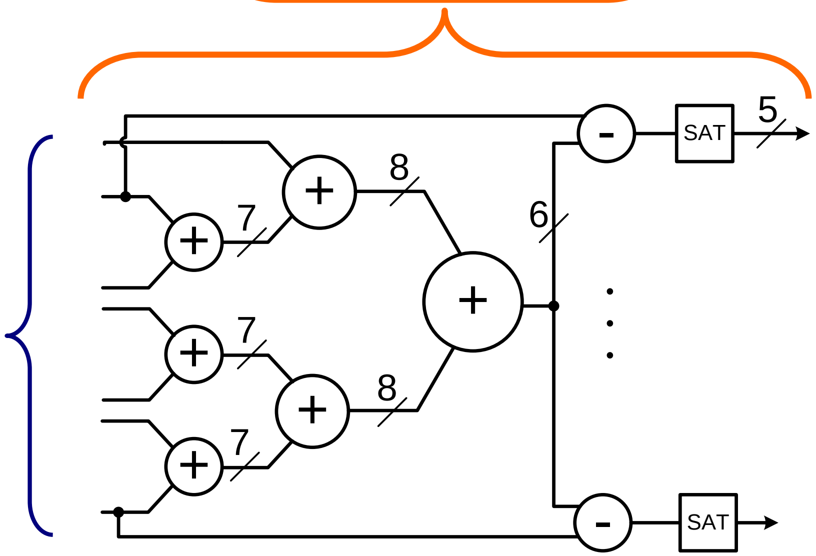

q_{1j} … q_{30j}
sort
min_1
min_2
compare-select

Variable Node Processor

$$\beta_{ij} = \sum_{j',h_{ij'}=1} \alpha_{ij'} + \lambda_j$$

- Based on the variable update equation
  - The same as the original MinSum and SPA algorithms
- Variable node hardware complexity is mainly reduced via wordwidth reduction

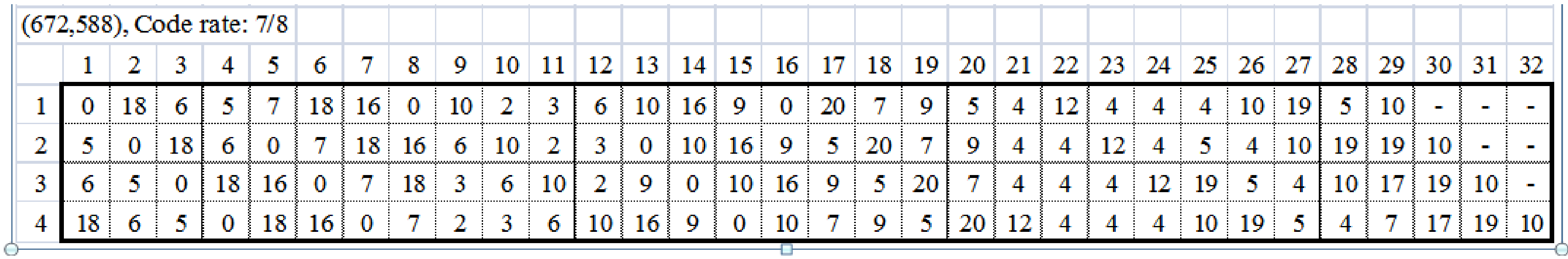




seven 5-bit inputs

Partial parallel decoder example

802.11ad LDPC code



(672,588), Code rate: 7/8

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	0	18	6	5	7	18	16	0	10	2	3	6	10	16	9	0	20	7	9	5	4	12	4	4	4	10	19	5	10	-	-	-
2	5	0	18	6	0	7	18	16	6	10	2	3	0	10	16	9	5	20	7	9	4	4	12	4	5	4	10	19	19	10	-	-
3	6	5	0	18	16	0	7	18	3	6	10	2	9	0	10	16	9	5	20	7	4	4	4	12	19	5	4	10	17	19	10	-
4	18	6	5	0	18	16	0	7	2	3	6	10	16	9	0	10	7	9	5	20	12	4	4	4	10	19	5	4	7	17	19	10

$$H=\left[\begin{array}{ccc|ccc|ccc|ccc}
0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
\hline
0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{array}\right]$$

Transmission scenario