Terabits-per-Second Throughput for Polar Codes

Altuğ Süral, E. Göksu Sezer, Yiğit Ertuğrul,
Orhan Arıkan and Erdal Arıkan

Bilkent University and Polaran Ltd.
Ankara, Turkey

Sep. 8th, 2019
IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2019)
Istanbul, Turkey

India's moon landing suffers last-minute communications loss

Prime minister Narendra Modi consoles scientists distraught as complex mission goes awry

▲ An ISRO employee reacts after the communication and data were lost from the Vikram Lander. Photograph: Jagadeesh Nv/EPA

Source:

https://www.theguardian.com/world/2019/sep/06/indias-moon-landing-suffers-last-minute-communications-loss

Why do we need Tb/s FEC?

- The 2018 Ethernet Roadmap foresees a demand for Tb/s data rates for 2020 and beyond [eth18]

Why do we need Tb/s FEC?

- The 2018 Ethernet Roadmap foresees a demand for Tb/s data rates for 2020 and beyond [eth18]
- Enable Tb/s wireless infrastructure for contemporary applications
  - fronthauling/backhauling
  - server clustering in data centers
  - drone-based communications
  - virtual and augmented reality
  - chip-to-chip and intra-chip communications
  - data transfer on data kiosks

Why do we need Tb/s FEC?

- The 2018 Ethernet Roadmap foresees a demand for Tb/s data rates for 2020 and beyond [eth18]
- Enable Tb/s wireless infrastructure for contemporary applications
  - fronthauling/backhauling
  - server clustering in data centers
  - drone-based communications
  - virtual and augmented reality
  - chip-to-chip and intra-chip communications
  - data transfer on data kiosks
- Contribute to the rapidly emerging standardization studies
  - IEEE 802.3ba Ethernet
  - IEEE 802.15.3d-2017
  - IEEE 802.11bb Light Communication TG (Li-Fi)

Why do we need Tb/s FEC?

- The 2018 Ethernet Roadmap foresees a demand for Tb/s data rates for 2020 and beyond [eth18]
- Enable Tb/s wireless infrastructure for contemporary applications
  - fronthauling/backhauling
  - server clustering in data centers
  - drone-based communications
  - virtual and augmented reality
  - chip-to-chip and intra-chip communications
  - data transfer on data kiosks
- Contribute to the rapidly emerging standardization studies
  - IEEE 802.3ba Ethernet
  - IEEE 802.15.3d-2017
  - IEEE 802.11bb Light Communication TG (Li-Fi)
- Provide strong error-resilience for those applications using **polar codes**

What are the requirements of Tb/s FEC?

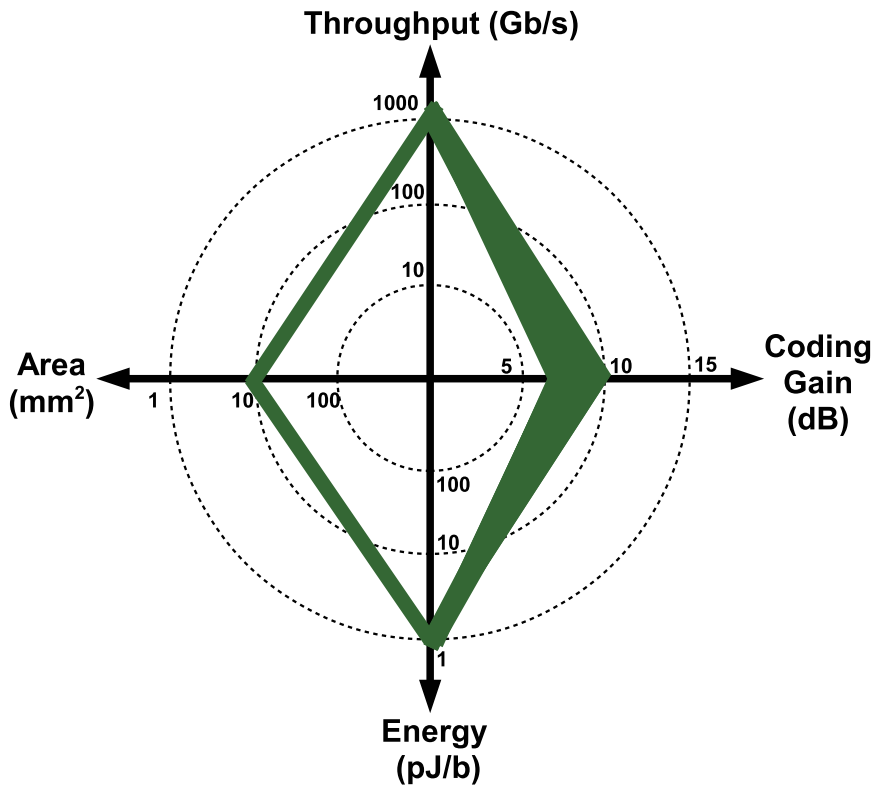
The ASIC design experts in H2020 EPIC project predicts Tb/s FEC throughput within the limits of KPIs [epi18].

Name of KPI	Requirement
Technology	7nm
Throughput	1 Tb/s
Clock freq.	≤ 1 GHz
Area	≤ 10 mm^2
Power	≤ 1 W
Power den.	≤ 0.1 W/mm^2
Area eff.	≥ 100 Gb/s/mm^2
Energy	≤ 1 pJ/bit

What are the requirements of Tb/s FEC?

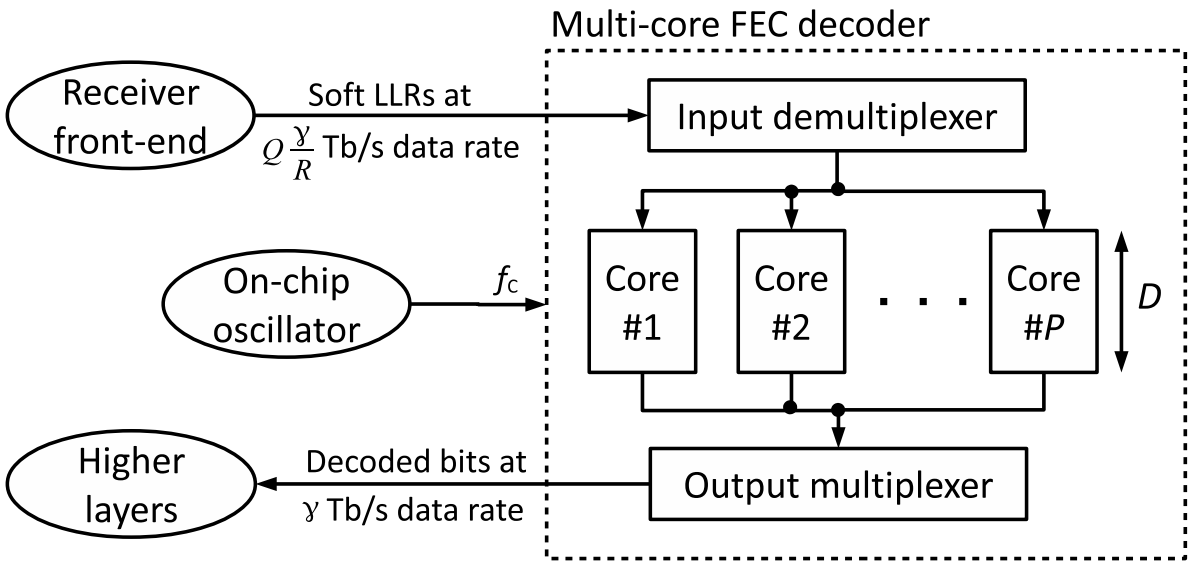
The ASIC design experts in H2020 EPIC project predicts Tb/s FEC throughput within the limits of KPIs [epi18].

Name of KPI	Requirement
Technology	7nm
Throughput	1 Tb/s
Clock freq.	≤ 1 GHz
Area	≤ 10 mm^2
Power	≤ 1 W
Power den.	≤ 0.1 W/mm^2
Area eff.	≥ 100 Gb/s/mm^2
Energy	≤ 1 pJ/bit

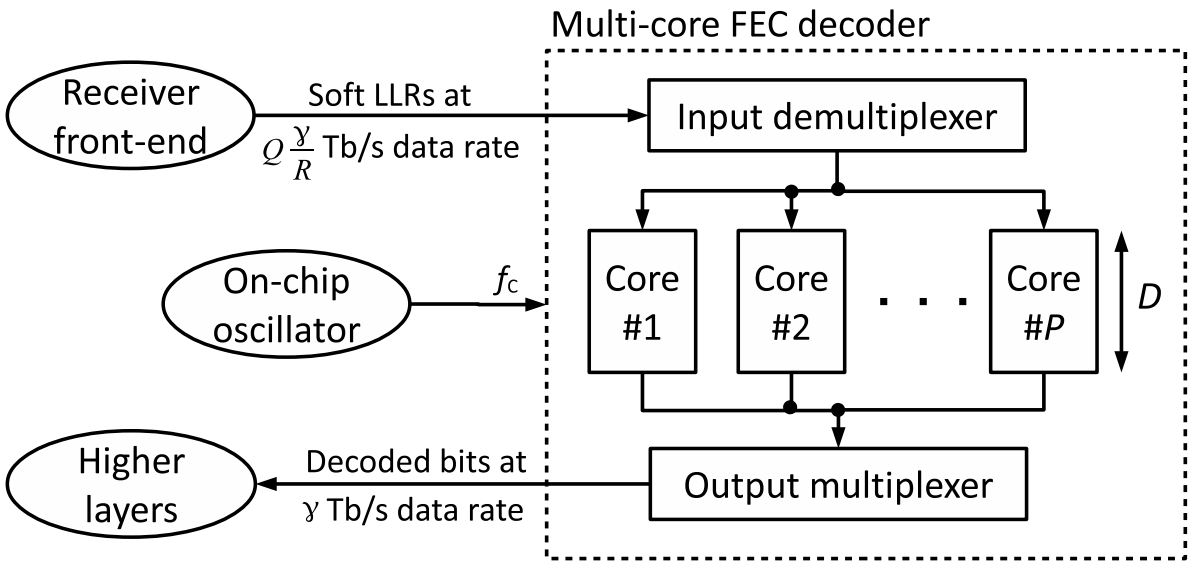


Challenges for Tb/s FEC

Challenges for Tb/s FEC

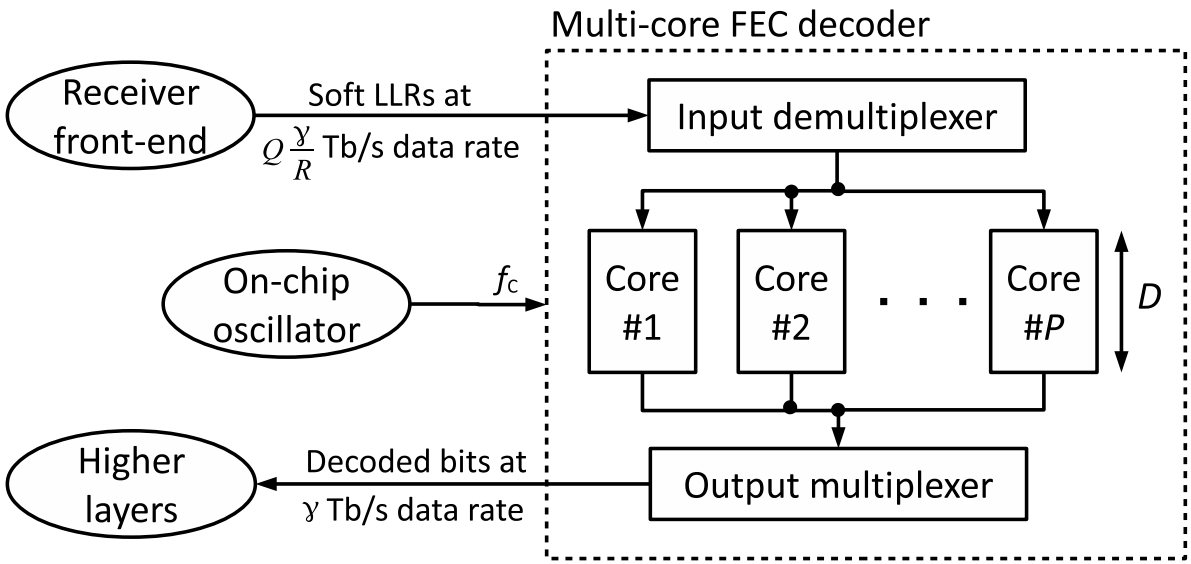


Challenges for Tb/s FEC



- Operating frequency (f_c) and throughput (γ) imbalance

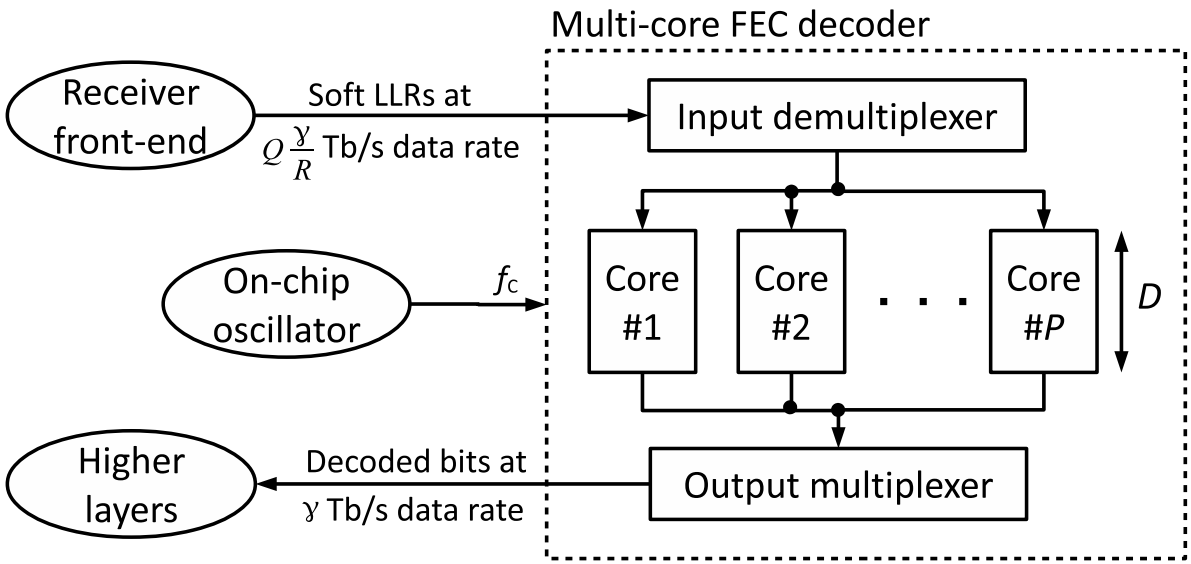
Challenges for Tb/s FEC



- Operating frequency (f_c) and throughput (γ) imbalance

For example, $\frac{\gamma}{f_c} = \frac{1 \text{ Tb/s}}{1 \text{ GHz}} = 1000$ bits

Challenges for Tb/s FEC

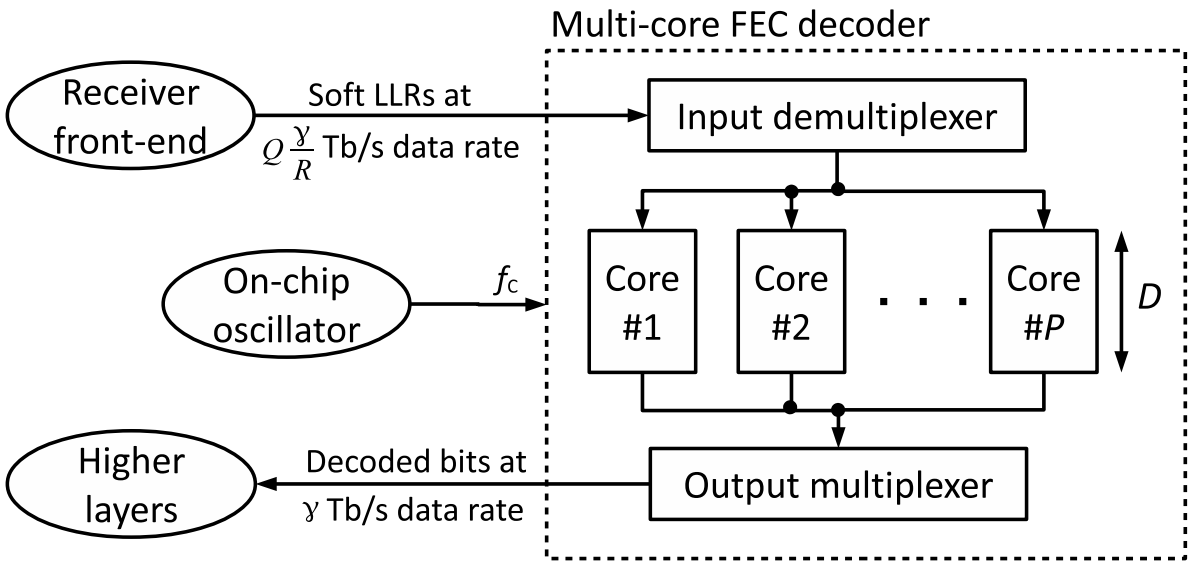


- Operating frequency (f_c) and throughput (γ) imbalance

  For example, $\frac{\gamma}{f_c} = \frac{1 \text{ Tb/s}}{1 \text{ GHz}} = 1000$ bits

- Data width (W) bottleneck

Challenges for Tb/s FEC



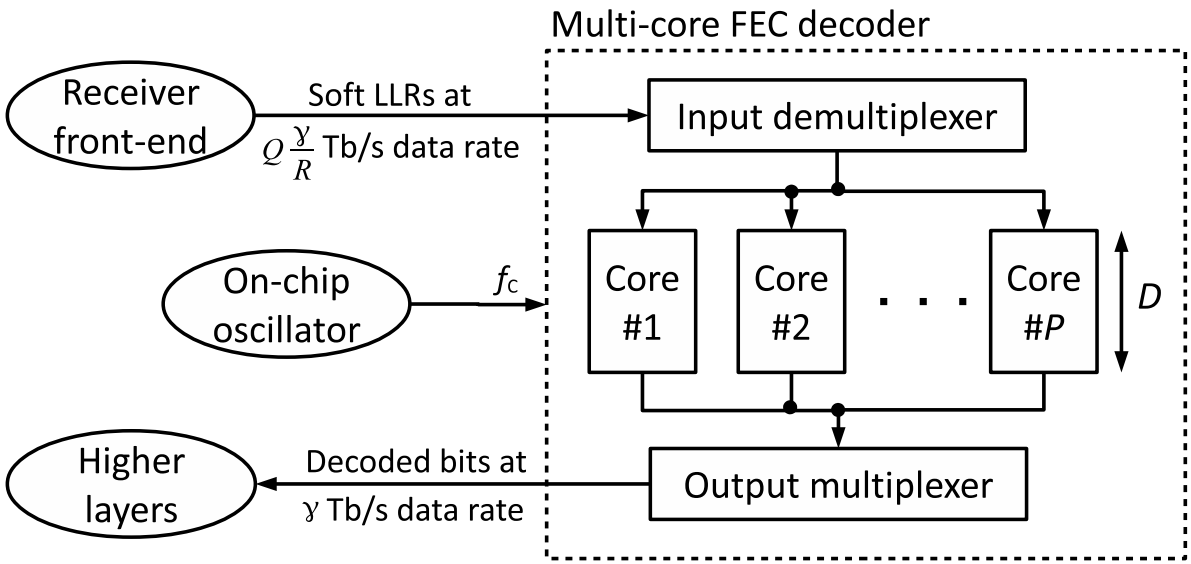
- Operating frequency (f_c) and throughput (γ) imbalance

  For example, $\frac{\gamma}{f_c} = \frac{1 \text{ Tb/s}}{1 \text{ GHz}} = 1000$ bits

- Data width (W) bottleneck

$$W = \frac{\gamma \times Q}{f_c \times R}$$

Challenges for Tb/s FEC



- Operating frequency (f_c) and throughput (γ) imbalance

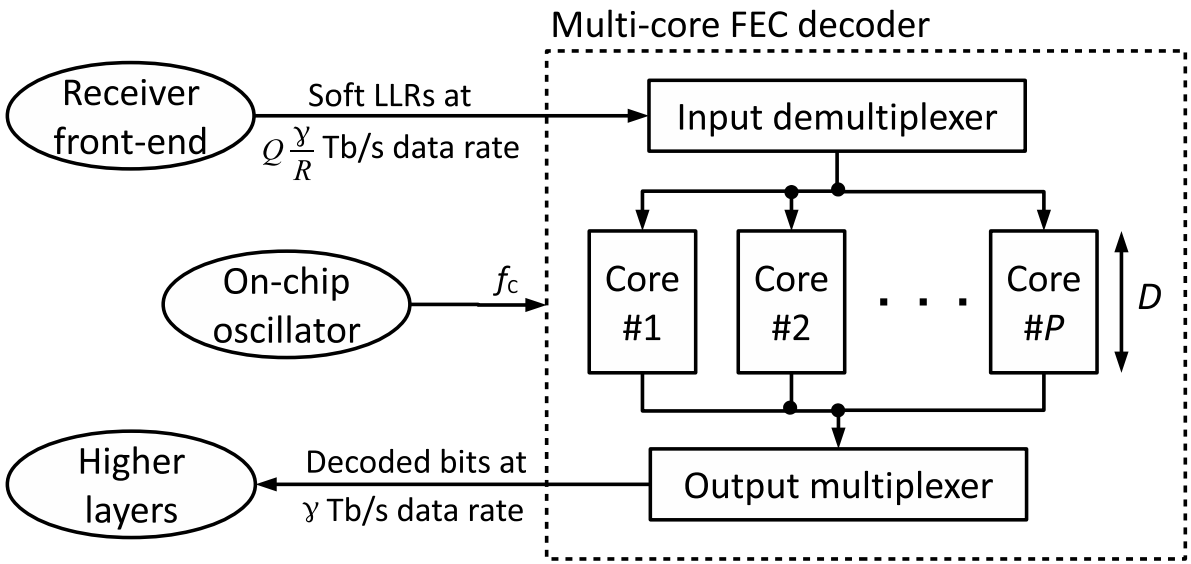
  For example, $\frac{\gamma}{f_c} = \frac{1 \text{ Tb/s}}{1 \text{ GHz}} = 1000$ bits

- Data width (W) bottleneck

$$W = \frac{\gamma \times Q}{f_c \times R}$$

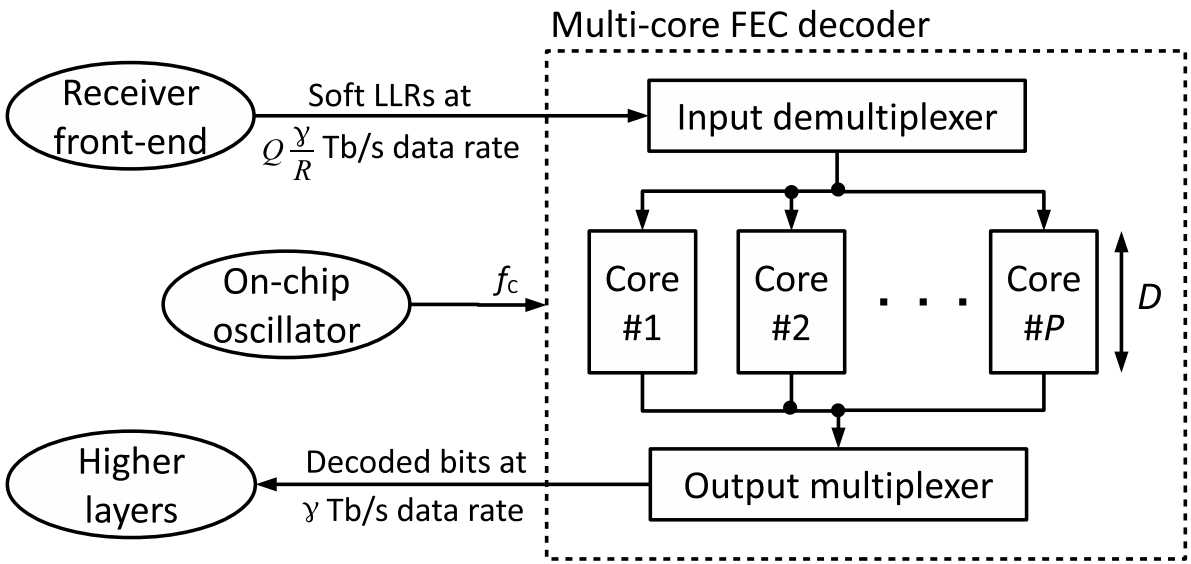
For example, $W = \frac{\gamma{=}1 \text{ Tb/s}}{f_c{=}1 \text{ GHz}} \times \frac{Q{=}3 \text{ bits}}{R{=}1/2} = 6000$ bits

Challenges for Tb/s FEC



- Excessive memory usage

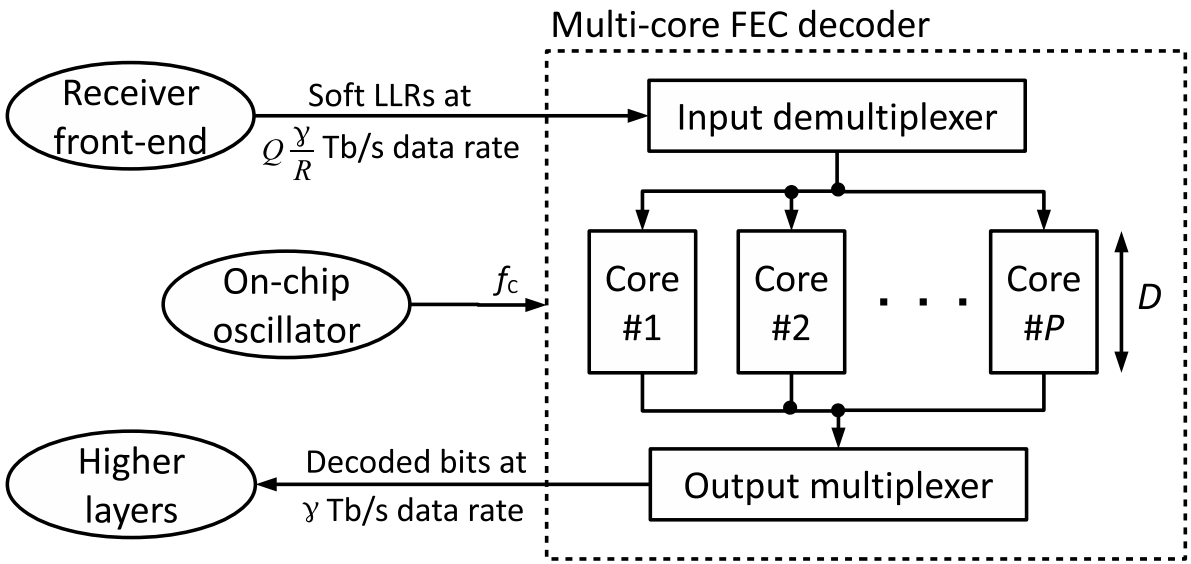
Challenges for Tb/s FEC



- Excessive memory usage

  There will be PD codewords inside the decoder at any moment

Challenges for Tb/s FEC

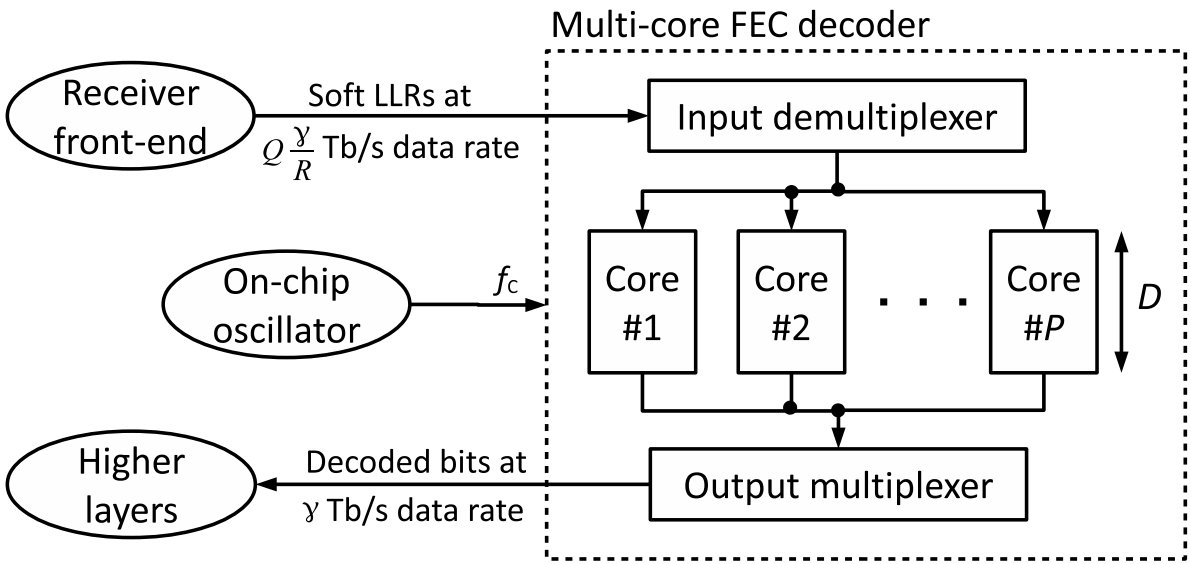


- Excessive memory usage

  There will be PD codewords inside the decoder at any moment

  Let Q' is average number of bits per LLR value inside decoder

Challenges for Tb/s FEC



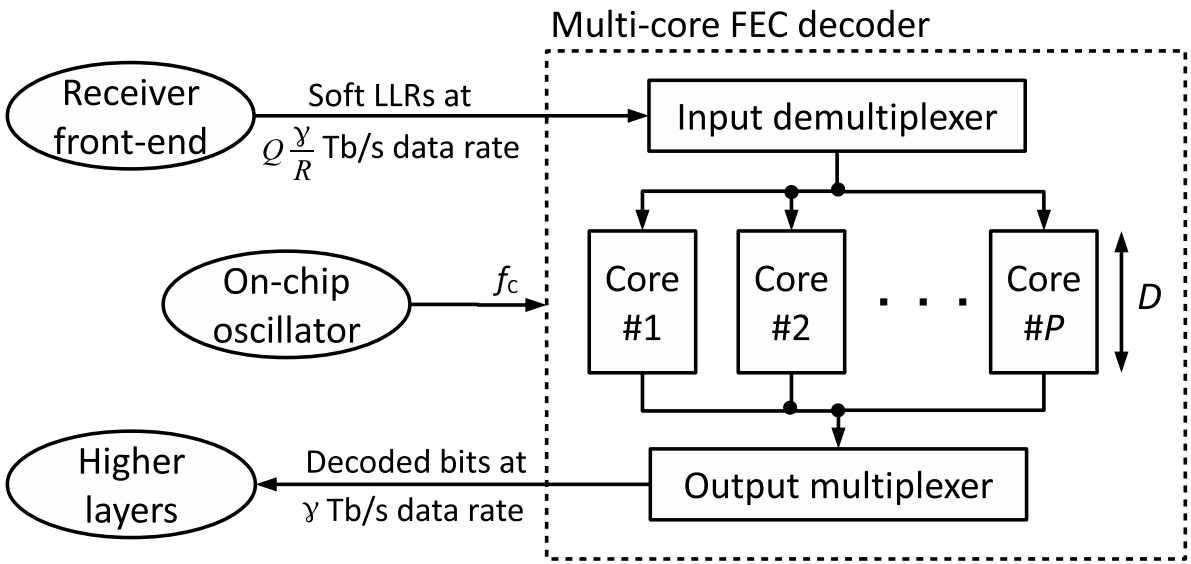
- Excessive memory usage

  There will be PD codewords inside the decoder at any moment

  Let Q' is average number of bits per LLR value inside decoder

  The memory requirement for this architecture is

Challenges for Tb/s FEC



- Excessive memory usage

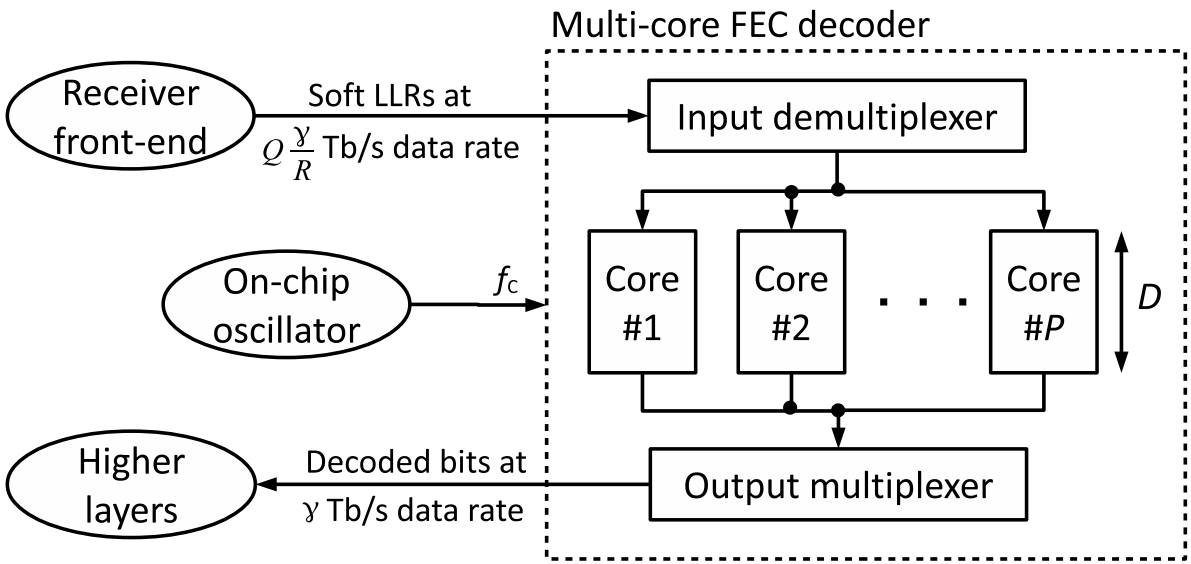
  There will be PD codewords inside the decoder at any moment

  Let Q' is average number of bits per LLR value inside decoder

  The memory requirement for this architecture is

$$M_{\text{Req}} = \frac{\gamma}{f_c}\frac{DQ'}{R} = \frac{NRPf_c}{f_c}\frac{DQ'}{R} = NPDQ'$$

Challenges for Tb/s FEC



- Excessive memory usage

  There will be PD codewords inside the decoder at any moment

  Let Q' is average number of bits per LLR value inside decoder

  The memory requirement for this architecture is

$$M_{\text{Req}} = \frac{\gamma}{f_c}\frac{DQ'}{R} = \frac{NRPf_c}{f_c}\frac{DQ'}{R} = NPDQ'$$

  In fully-parallel successive cancellation (SC) polar decoder,

$$D = 2N - 2$$

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]
- Use Majority-Logic aided Successive Cancellation (SC-MJL) decoding algorithm in [Diz17] with several enhancements

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]
- Use Majority-Logic aided Successive Cancellation (SC-MJL) decoding algorithm in [Diz17] with several enhancements
- Decode the repetition (REP) and single parity-check (SPC) code segments with MAP [FMI99] and Wagner [SB54] decoders respectively (similar to [SGV+14] and [HA17])

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]
- Use Majority-Logic aided Successive Cancellation (SC-MJL) decoding algorithm in [Diz17] with several enhancements
- Decode the repetition (REP) and single parity-check (SPC) code segments with MAP [FMI99] and Wagner [SB54] decoders respectively (similar to [SGV+14] and [HA17])
- Develop a dedicated hardware architecture for the SC-MJL algorithm

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]
- Use Majority-Logic aided Successive Cancellation (SC-MJL) decoding algorithm in [Diz17] with several enhancements
- Decode the repetition (REP) and single parity-check (SPC) code segments with MAP [FMI99] and Wagner [SB54] decoders respectively (similar to [SGV+14] and [HA17])
- Develop a dedicated hardware architecture for the SC-MJL algorithm
  - Unlike [Diz17], here we focus on throughput only and use a fully unrolled and pipelined architecture

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]
- Use Majority-Logic aided Successive Cancellation (SC-MJL) decoding algorithm in [Diz17] with several enhancements
- Decode the repetition (REP) and single parity-check (SPC) code segments with MAP [FMI99] and Wagner [SB54] decoders respectively (similar to [SGV+14] and [HA17])
- Develop a dedicated hardware architecture for the SC-MJL algorithm
  - Unlike [Diz17], here we focus on throughput only and use a fully unrolled and pipelined architecture
  - Dedicated hardware modules for each set of operation

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]
- Use Majority-Logic aided Successive Cancellation (SC-MJL) decoding algorithm in [Diz17] with several enhancements
- Decode the repetition (REP) and single parity-check (SPC) code segments with MAP [FMI99] and Wagner [SB54] decoders respectively (similar to [SGV+14] and [HA17])
- Develop a dedicated hardware architecture for the SC-MJL algorithm
  - Unlike [Diz17], here we focus on throughput only and use a fully unrolled and pipelined architecture
  - Dedicated hardware modules for each set of operation
- Implement a variable-length quantization scheme inside the decoder

Polar codes for Tb/s FEC

- Develop a specific solution based on systematic polar codes [Ari09], [Ari11]
- Use Majority-Logic aided Successive Cancellation (SC-MJL) decoding algorithm in [Diz17] with several enhancements
- Decode the repetition (REP) and single parity-check (SPC) code segments with MAP [FMI99] and Wagner [SB54] decoders respectively (similar to [SGV+14] and [HA17])
- Develop a dedicated hardware architecture for the SC-MJL algorithm
  - Unlike [Diz17], here we focus on throughput only and use a fully unrolled and pipelined architecture
  - Dedicated hardware modules for each set of operation
- Implement a variable-length quantization scheme inside the decoder
- Merge pipelined decoding stages for register balancing/retiming

Proposed SC-MJL decoding algorithm

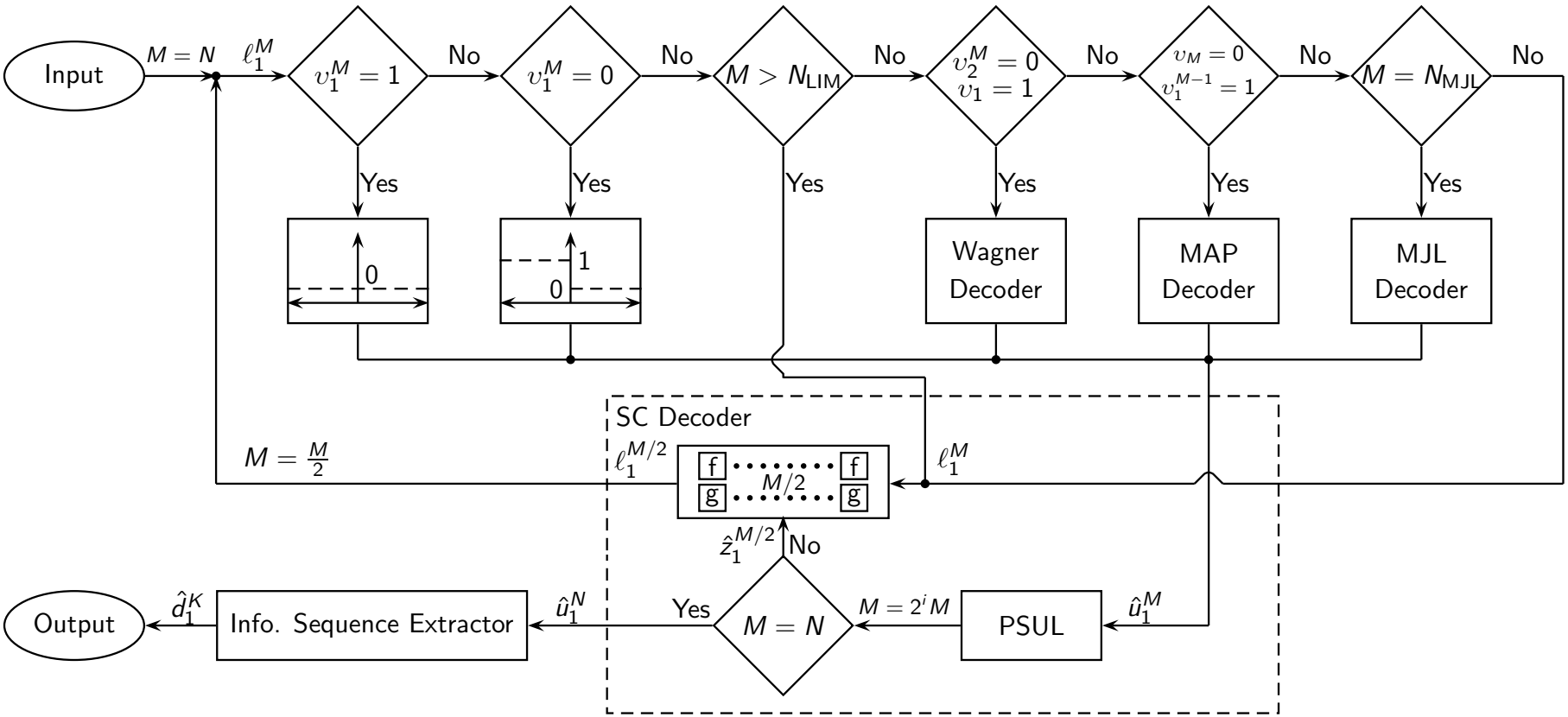
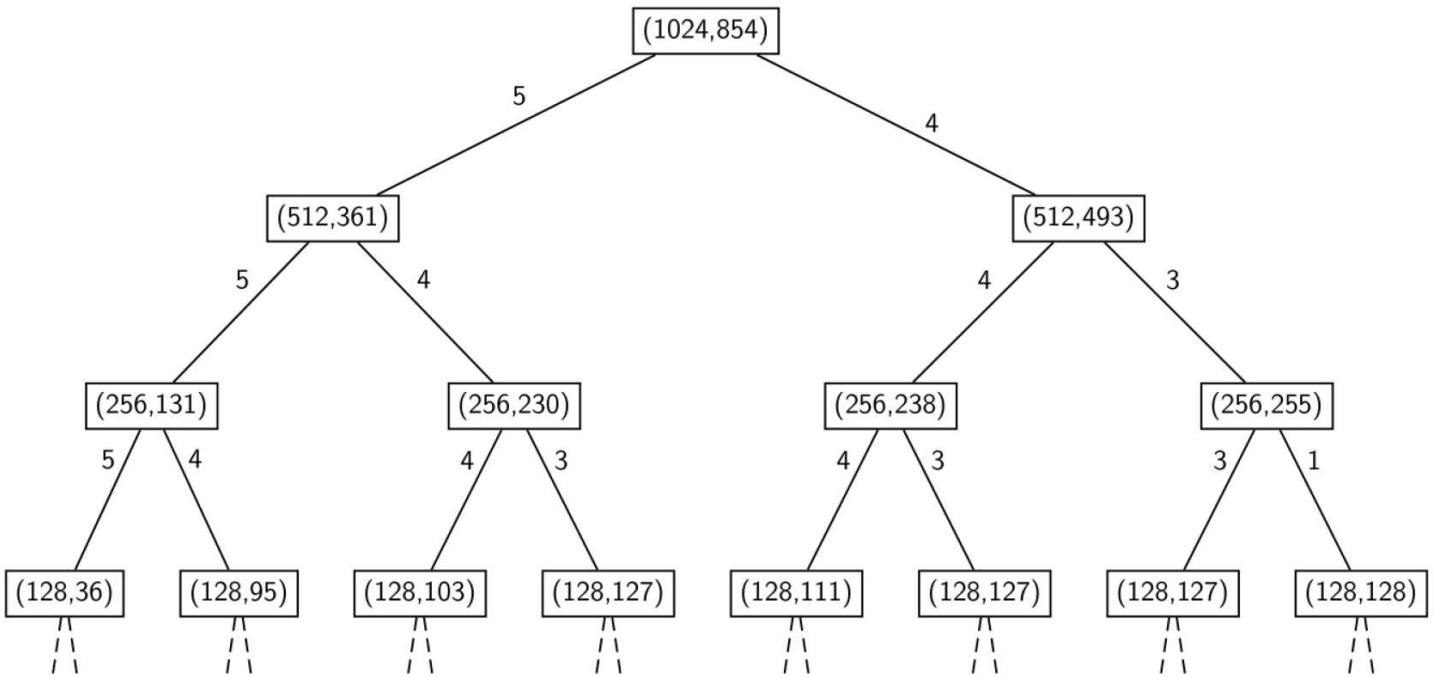


Figure: Data flowchart of SC-MJL decoding algorithm where N is the code block length, K is the number of information bits, v_1^M is the indicator vector of frozen bits with variable constituent block length M, N_{MJL} is the block length of MJL decoder and N_{LIM} is the maximum block length of Wagner and MAP decoders.

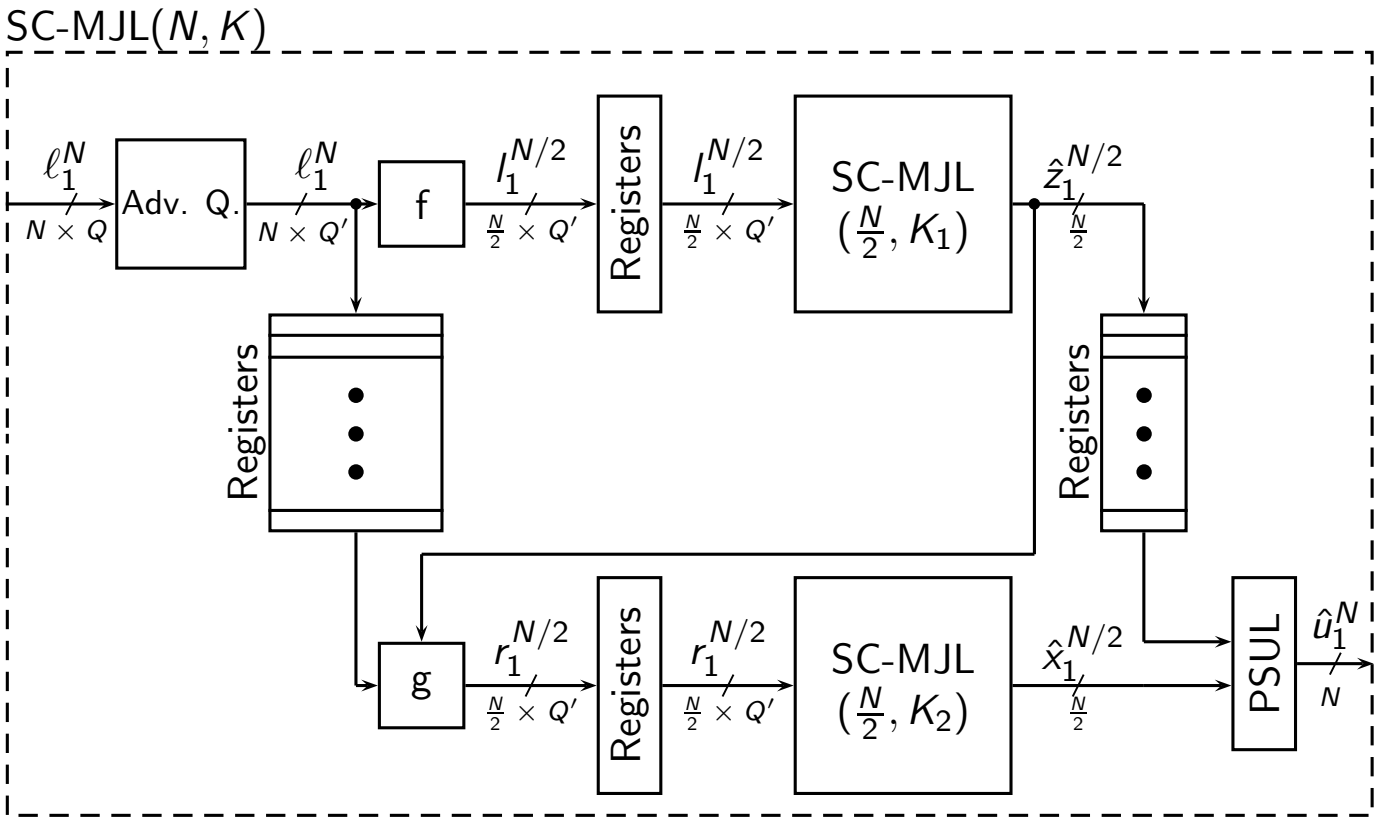
Adaptive quantization of LLRs inside the decoder



- The quantization bits are optimized using input LLR distribution of each polar code segment
- Since polarization takes place, using large number of bits is not necessary for the polarized code segments

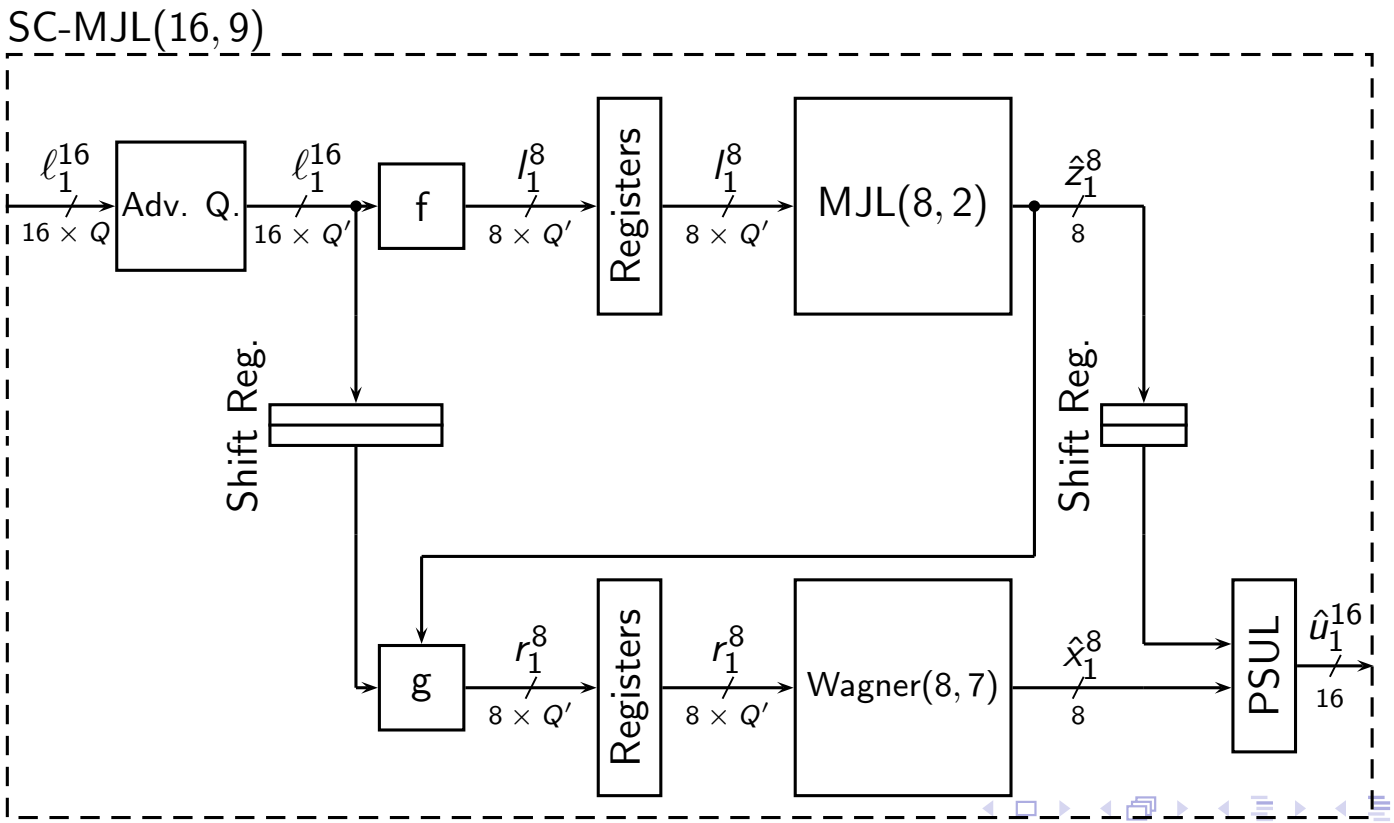
Proposed hardware architecture for SC-MJL

- For an arbitrary block length N, number of information bits K, input quantization Q and internal quantization Q', the SC-MJL architecture is



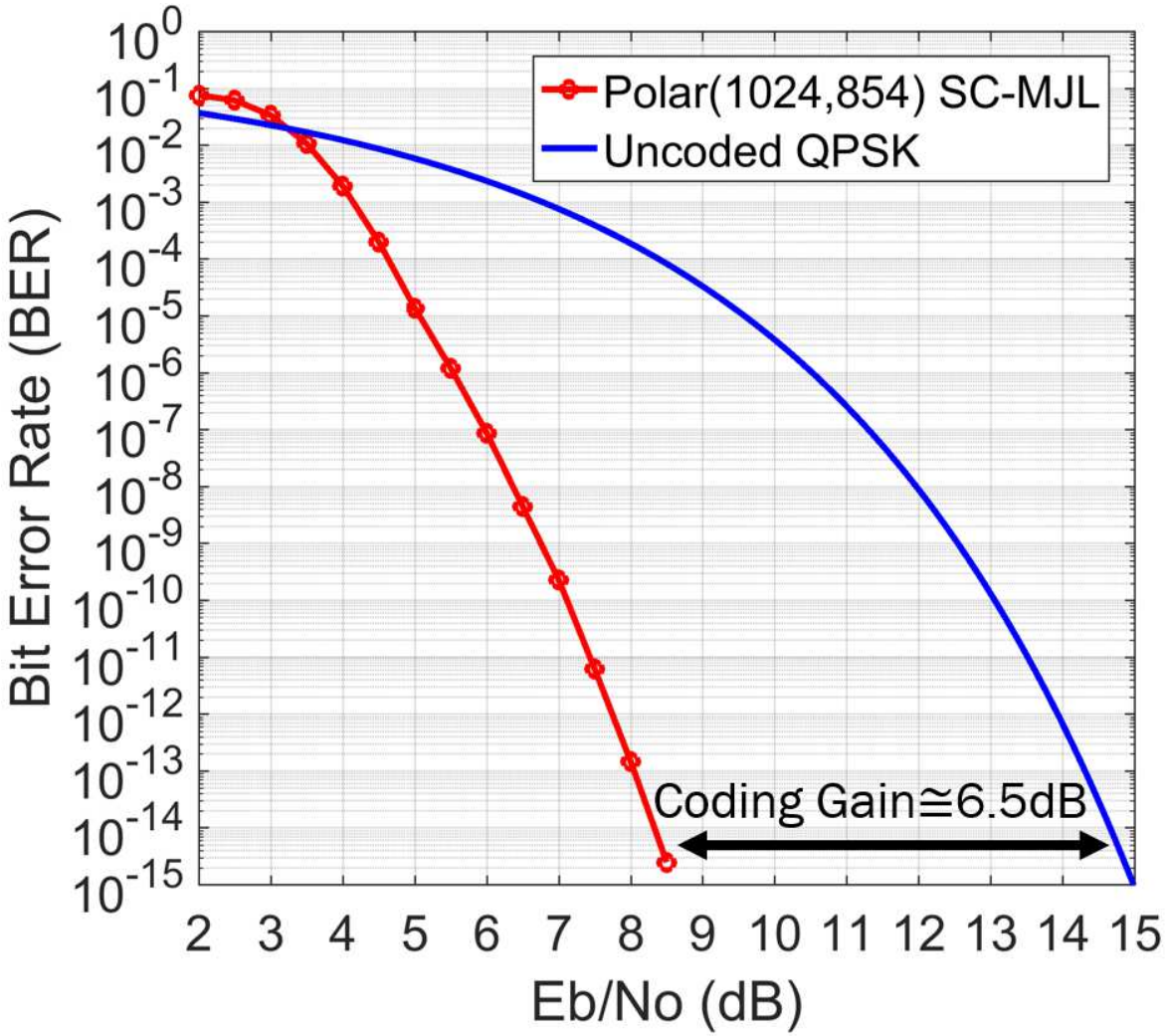
Proposed hardware architecture for SC-MJL

- An example $N = 16$, $K = 9$ design with $N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$ parameters
- Use MJL when $v_1^8 = \{1, 1, 1, 0, 1, 1, 1, 0\}$
- Use Wagner when $v_1^8 = \{1, 0, 0, 0, 0, 0, 0, 0\}$



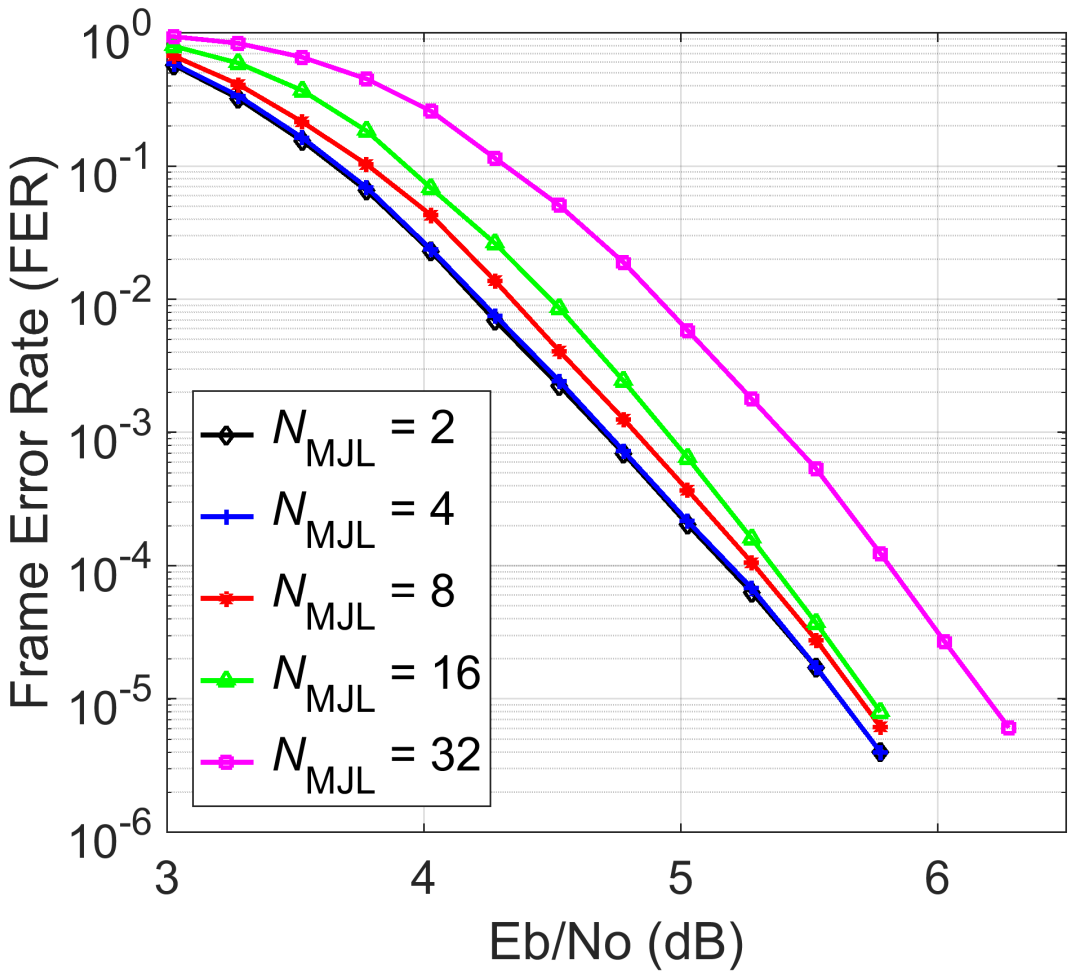
FPGA performance of SC-MJL decoder

- Carried out with an AWGN channel



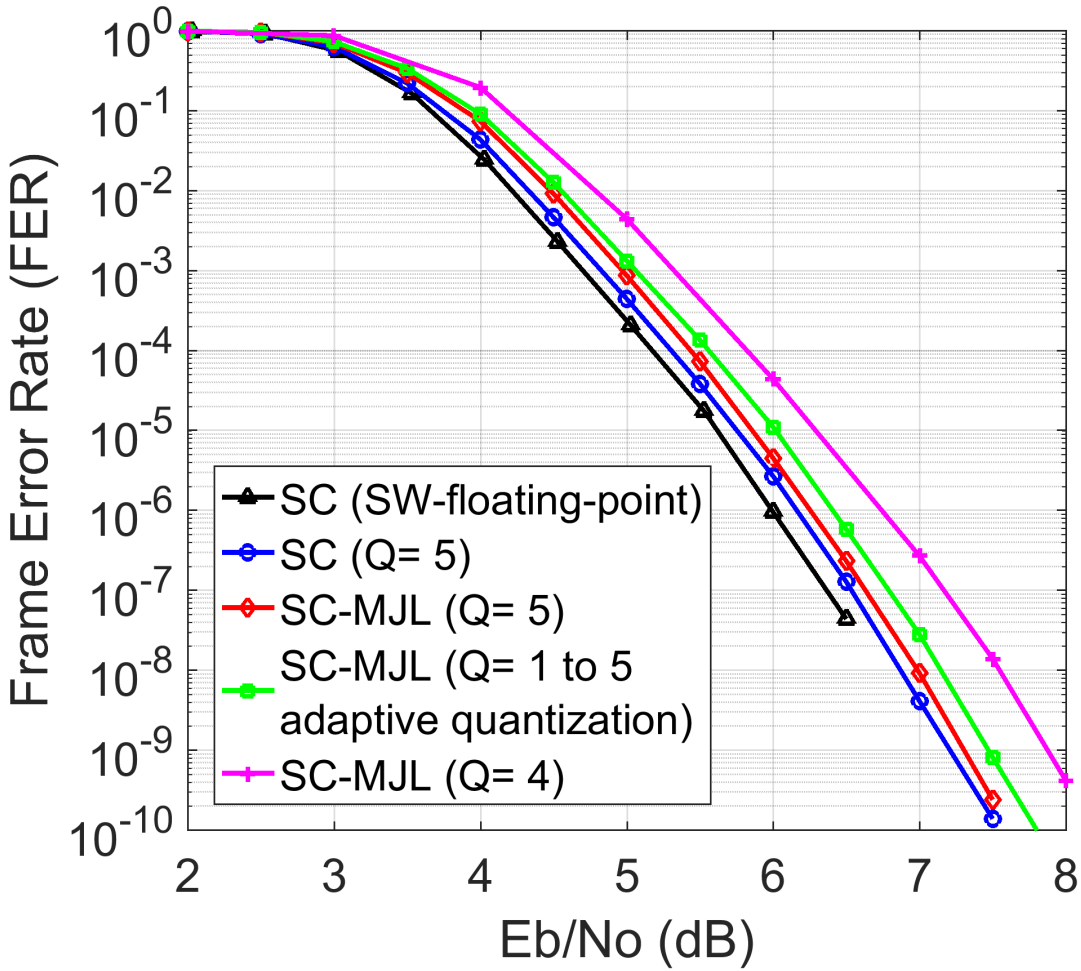
FPGA performance of SC-MJL decoder

- As N_{MJL} increases the performance deteriorates progressively



FPGA performance comparison

- Observed tolerable performance loss under adaptive quantization



ASIC 45nm post-synthesis results of (1024,854) polar code

$N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing Arch.	x	x	x	✓
Throughput (Gb/s)			427	
Frequency (MHz)			500	

ASIC 45nm post-synthesis results of (1024,854) polar code

$N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing Arch.	x	x	x	✓
Throughput (Gb/s)			427	
Frequency (MHz)			500	
Area (mm^2)	9.8			
Area Eff. ($Gb/s/mm^2$)	43.5			
Power (W)	4.6			
Pow. Den. (W/mm^2)	0.47			
Energy Eff. (pJ/bit)	10.9			
Latency (μs)	0.31			
Latency (Clock cycles)	157			

ASIC 45nm post-synthesis results of (1024,854) polar code

$N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing Arch.	x	x	x	✓
Throughput (Gb/s)			427	
Frequency (MHz)			500	
Area (mm^2)	9.8	$\xrightarrow{15\%}$ 8.3		
Area Eff. ($Gb/s/mm^2$)	43.5	51.4		
Power (W)	4.6	3.1		
Pow. Den. (W/mm^2)	0.47	0.38		
Energy Eff. (pJ/bit)	10.9	$\xrightarrow{33\%}$ 7.3		
Latency (μs)	0.31	0.25		
Latency (Clock cycles)	157	$\xrightarrow{19\%}$ 127		

ASIC 45nm post-synthesis results of (1024,854) polar code

$N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing Arch.	x	x	x	✓
Throughput (Gb/s)			427	
Frequency (MHz)			500	
Area (mm^2)	9.8	8.3	$\xrightarrow{20\%}$	6.6
Area Eff. ($Gb/s/mm^2$)	43.5	51.4		65.0
Power (W)	4.6	3.1		2.3
Pow. Den. (W/mm^2)	0.47	0.38		0.36
Energy Eff. (pJ/bit)	10.9	7.3	$\xrightarrow{33\%}$	5.5
Latency (μs)	0.31	0.25		0.25
Latency (Clock cycles)	157	127		127

ASIC 45nm post-synthesis results of (1024,854) polar code

$N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing Arch.	x	x	x	✓
Throughput (Gb/s)			427	
Frequency (MHz)			500	
Area (mm^2)	9.8	15% → 8.3	20% → 6.6	64% → 2.4
Area Eff. ($Gb/s/mm^2$)	43.5	51.4	65.0	175.2
Power (W)	4.6	3.1	2.3	1.0
Pow. Den. (W/mm^2)	0.47	0.38	0.36	0.42
Energy Eff. (pJ/bit)	10.9	33% → 7.3	33% → 5.5	56% → 2.4
Latency (μs)	0.31	0.25	0.25	0.08
Latency (Clock cycles)	157	19% → 127	127	69% → 40

Scaled 7nm results in ASIC

- Use conservative scaling rules in [epi18] from 45nm to 7nm
- Use two identical polar decoders in parallel ($P = 2$)
- Multiply the expected 2.2 GHz clock freq. with 0.27

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing Arch.	x	x	x	✓
Area Scaling	14.3	16.9	21.4	57.7
Throughput (Gb/s)			1000	
Frequency (MHz)			585.5	
Area (mm^2)			10	
Area Eff. ($Gb/s/mm^2$)			100	
Power (W)	1.69	1.14	0.85	0.37
Pow. Den. (W/mm^2)	0.17	0.11	0.09	0.04
Energy Eff. (pJ/bit)	1.69	1.14	0.85	0.37

Comparison with other implementations

Implementation	This work	[GTG17]	[Diz17]
Architecture	SC-MJL	SC-Fast	SC-Comb.
ASIC Technology	45nm	28nm	90nm
Coded Throughput (Gb/s)	512	1275	2.6
Frequency (MHz)	500	1245	2.5
Area (mm^2)	2.4	4.6	3.2
Power (W)	1.01	8.79	0.19
Converted to 28nm, 1.0 V using the scaling in [WC10], [Diz17]			
Coded Throughput (Gb/s)	823	1275	8.2
Area (mm^2)	0.94^a	4.63	0.31
Area Eff. ($Gb/s/mm^2$)	872	276	26
Power (W)	$0.44^\dagger$	8.79	0.04
Energy Eff. (pJ/bit)	$0.5^\ddagger$	6.9	4.6

[a] Normalized factor for area is $0.39 = (28/45)^2$

[†] Norm. factor for power is $0.43 = (28/45)(1.0/1.2)^2$

[‡] Norm. factor for energy eff. is $0.27 = (28/45)^2(1.0/1.2)^2$

Summary

- 1 Tb/s polar coding scheme is feasible at 7nm with 6.5 dB coding gain

Summary

- 1 Tb/s polar coding scheme is feasible at 7nm with 6.5 dB coding gain
- List of techniques used in the proposed solution
  - SC decoding in initial stages where parallelism is high
  - Use shortcuts for easily decodable code segments
  - MJL decoding for making parallel decisions
  - Adaptive quantization to reduce memory usage
  - Register balancing to reduce pipeline depth

Summary

- 1 Tb/s polar coding scheme is feasible at 7nm with 6.5 dB coding gain
- List of techniques used in the proposed solution
  - SC decoding in initial stages where parallelism is high
  - Use shortcuts for easily decodable code segments
  - MJL decoding for making parallel decisions
  - Adaptive quantization to reduce memory usage
  - Register balancing to reduce pipeline depth
- Final remarks

Summary

- 1 Tb/s polar coding scheme is feasible at 7nm with 6.5 dB coding gain
- List of techniques used in the proposed solution
  - SC decoding in initial stages where parallelism is high
  - Use shortcuts for easily decodable code segments
  - MJL decoding for making parallel decisions
  - Adaptive quantization to reduce memory usage
  - Register balancing to reduce pipeline depth
- Final remarks
  - Main challenges are frequency/throughput imbalance, I/O bottleneck and storage complexity

Summary

- 1 Tb/s polar coding scheme is feasible at 7nm with 6.5 dB coding gain
- List of techniques used in the proposed solution
  - SC decoding in initial stages where parallelism is high
  - Use shortcuts for easily decodable code segments
  - MJL decoding for making parallel decisions
  - Adaptive quantization to reduce memory usage
  - Register balancing to reduce pipeline depth
- Final remarks
  - Main challenges are frequency/throughput imbalance, I/O bottleneck and storage complexity
  - SC-MJL decoder achieves 427 Gb/s at 45nm technology

Summary

- 1 Tb/s polar coding scheme is feasible at 7nm with 6.5 dB coding gain
- List of techniques used in the proposed solution
  - SC decoding in initial stages where parallelism is high
  - Use shortcuts for easily decodable code segments
  - MJL decoding for making parallel decisions
  - Adaptive quantization to reduce memory usage
  - Register balancing to reduce pipeline depth
- Final remarks
  - Main challenges are frequency/throughput imbalance, I/O bottleneck and storage complexity
  - SC-MJL decoder achieves 427 Gb/s at 45nm technology
  - SC-MJL can achieve 1 Tb/s at 7nm under a feasible power and area budget

Summary

- 1 Tb/s polar coding scheme is feasible at 7nm with 6.5 dB coding gain
- List of techniques used in the proposed solution
  - SC decoding in initial stages where parallelism is high
  - Use shortcuts for easily decodable code segments
  - MJL decoding for making parallel decisions
  - Adaptive quantization to reduce memory usage
  - Register balancing to reduce pipeline depth
- Final remarks
  - Main challenges are frequency/throughput imbalance, I/O bottleneck and storage complexity
  - SC-MJL decoder achieves 427 Gb/s at 45nm technology
  - SC-MJL can achieve 1 Tb/s at 7nm under a feasible power and area budget
  - Main advantages are energy efficiency and area efficiency

Acknowledgements

- This work has been carried out by a support from the EPIC project with the funding from the European Union's Horizon 2020 research and innovation programme under grant No. 760150.

E. Arikan.

Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels.
IEEE Transactions on Information Theory, 55(7):3051–3073, July 2009.

E. Arikan.

Systematic polar coding.
IEEE Communications Letters, 15(8):860–862, August 2011.

Onur Dizdar.

High Throughput Decoding Methods and Architectures for Polar Codes with High Energy-Efficiency and Low Latency.
PhD thesis, Bilkent University, 2017.

EPIC - Enabling practical wireless Tb/s communications with next generation channel coding, 2018.
[Online]. Available: https://epic-h2020.eu/results.

Ethernet roadmap 2018, 2018.

[Online]. Available: https://ethernetalliance.org/wp-content/uploads/2016/03/EthernetRoadmap-2018-Side1-1600x1200.jpg.

M. P. C. Fossorier, M. Mihaljevic, and H. Imai.

Reduced complexity iterative decoding of low-density parity check codes based on belief propagation.
IEEE Trans. on Comm., 47(5):673–680, May. 1999.

Pascal Giard, Claude Thibeault, and Warren J. Gross.

High-Speed Decoders for Polar Codes.
Springer, 2017.
pp. 66–67.

M. Hanif and M. Ardakani.

Fast successive-cancellation decoding of polar codes: Identification and decoding of new nodes.
IEEE Communications Letters, 21(11):2360–2363, Nov 2017.

R. Silverman and M. Balser.

Coding for constant-data-rate systems.
Transactions of the IRE Professional Group on Information Theory, 4(4):50–63, September 1954.

G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross.

Fast polar decoders: Algorithm and implementation.
IEEE Journal on Selected Areas in Communications, 32(5):946–957, May 2014.

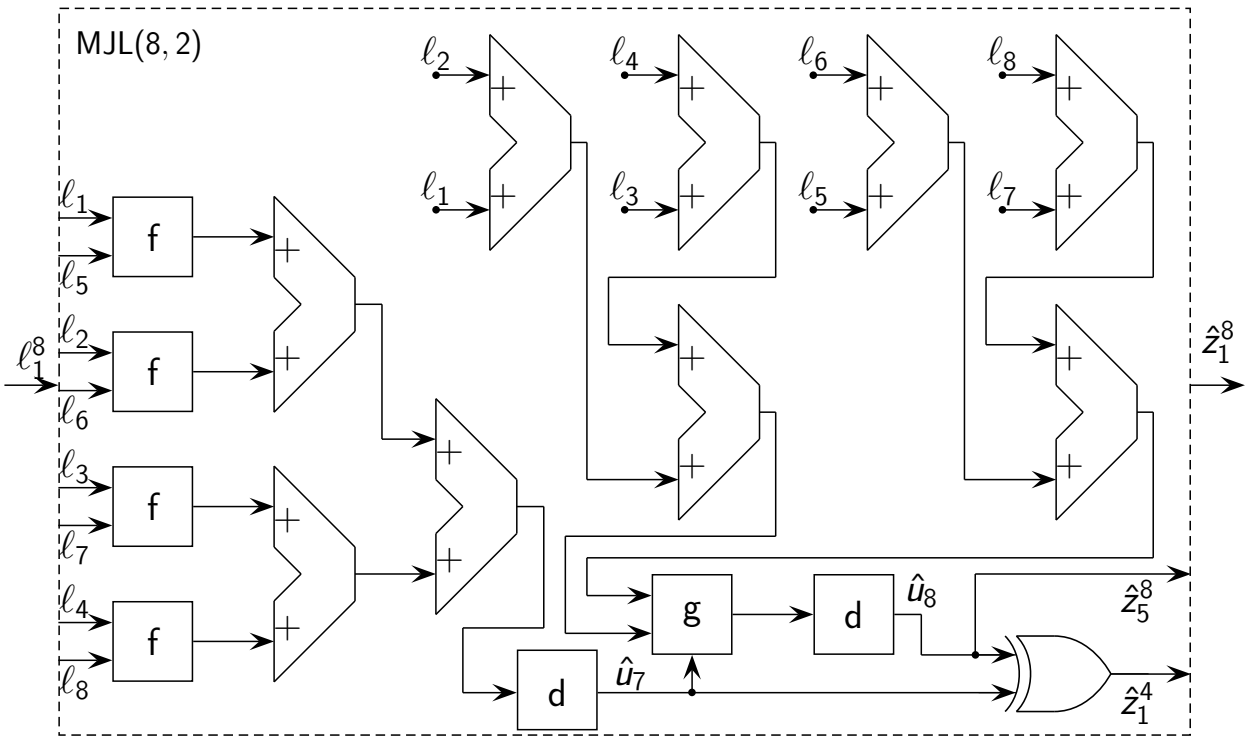
C. Wong and H. Chang.

Reconfigurable turbo decoder with parallel architecture for 3gpp lte system.
IEEE Transactions on Circuits and Systems II: Express Briefs, 57(7):566–570, July 2010.

Thank you for your attention!

Extra slides: design details

- MJL(8,2) has nine adders, four f functions, two d functions, one g function and one XOR gate



Extra slides: design details

- Each f function contains a comparator and an XOR gate
- Each g function has two adders and one multiplexer