83: A 0.75mm^2 407μW real-time speech audio denoiser with quantized cascaded redundant convolutional encoder-decoder for wearable IoT devices

Dimple Vijay Kochar, Maitreyi Ashok, Anantha P. Chandrakasan

Electrical Engineering and Computer Science
Massachusetts Institute of Technology, Cambridge, MA

Outline

- Introduction
- Design Features
  - Algorithm Design
  - Quantization Scheme
  - Top-level Chip Architecture
  - 1D Convolution Dataflow
- Results
- Conclusion

Outline

- **Introduction**
- Design Features
  - Algorithm Design
  - Quantization Scheme
  - Top-level Chip Architecture
  - 1D Convolution Dataflow
- Results
- Conclusion

Introduction

Growing Need for Audio Denoising in Wearable IoT Devices

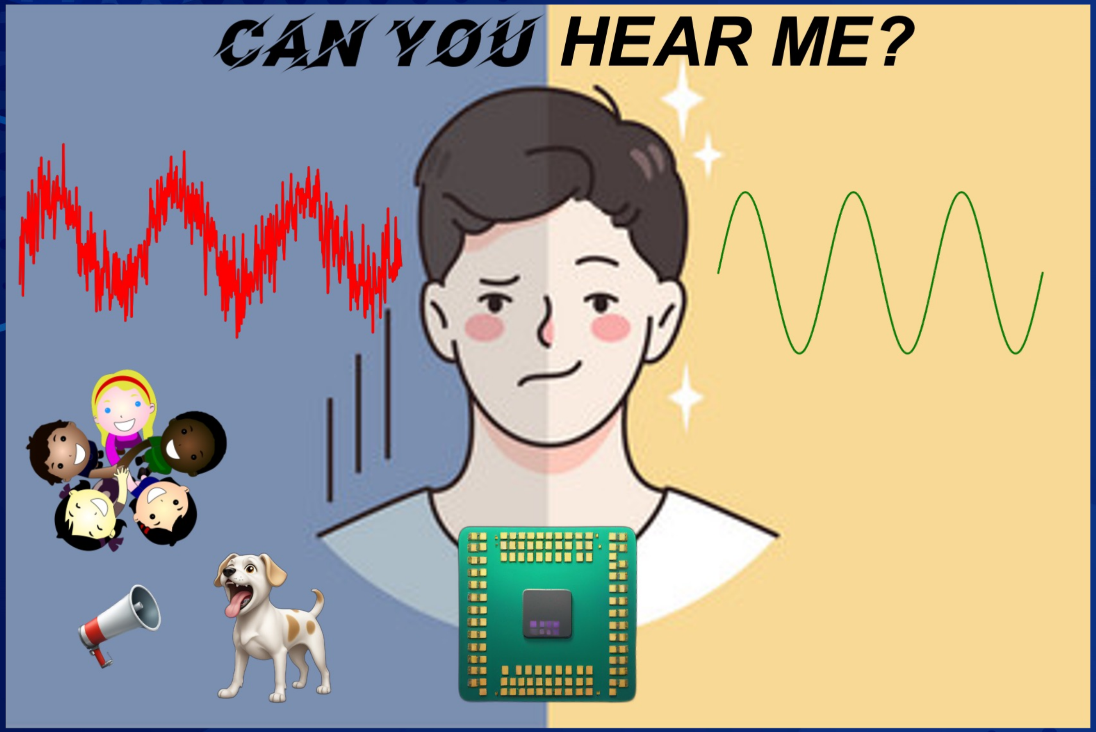
Introduction

Growing Need for Audio Denoising in Wearable IoT Devices

- Wearable IoT devices require effective audio denoising
  - Clear communication during calls
  - High-quality audio recordings
  - Enhanced voice assistants

Introduction

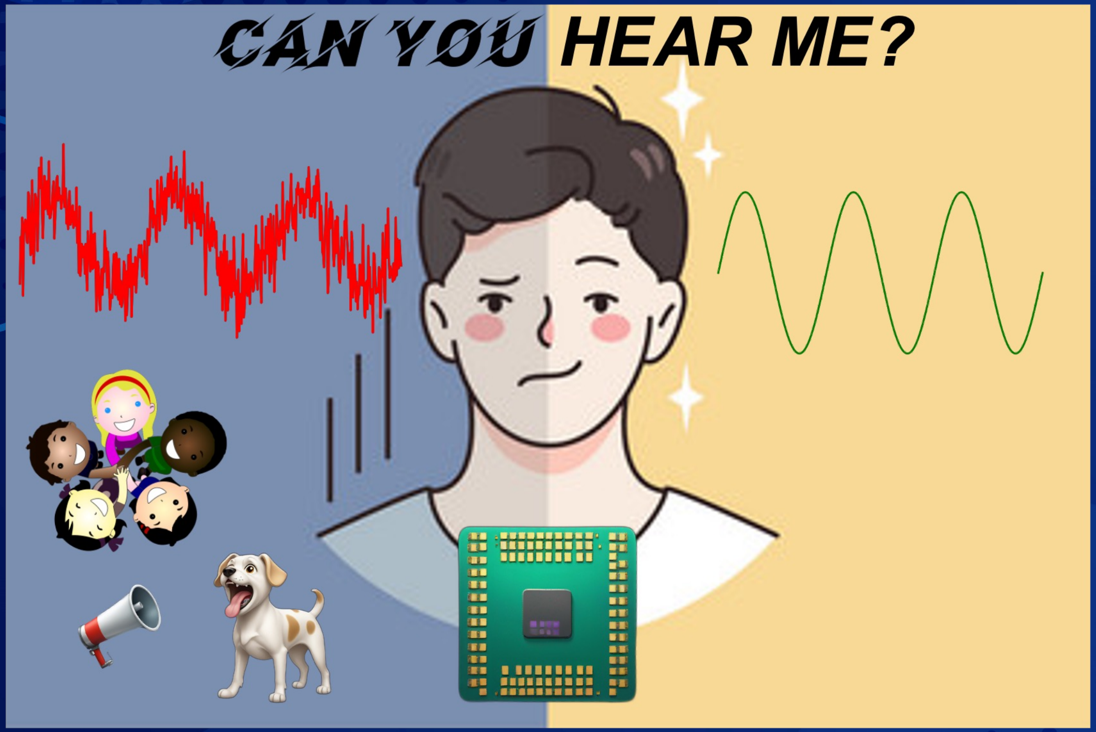
Growing Need for Audio Denoising in Wearable IoT Devices



- Wearable IoT devices require effective audio denoising
  - Clear communication during calls
  - High-quality audio recordings
  - Enhanced voice assistants

Introduction

Growing Need for Audio Denoising in Wearable IoT Devices



- Wearable IoT devices require effective audio denoising
  - Clear communication during calls
  - High-quality audio recordings
  - Enhanced voice assistants

- Audio denoising is a complex task involving *audio reconstruction*

Introduction

Audio Denoising is Hard for Wearable IoT Devices

https://ccrma.stanford.edu/~njb/teaching/sstutorial/
Park, Se Rim, and Jinwon Lee. arXiv preprint arXiv:1609.07132 (2016).

Introduction

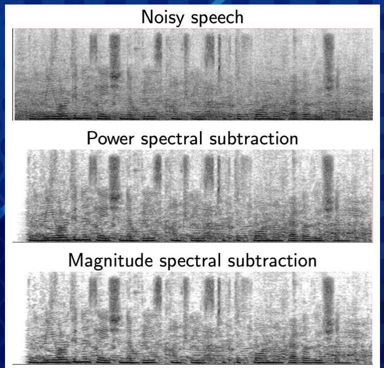
Audio Denoising is Hard for Wearable IoT Devices

- Wearables require:
  - Superior audio quality
  - Low power consumption
  - Realtime performance

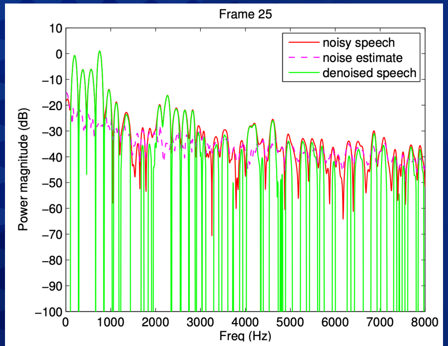
https://ccrma.stanford.edu/~njb/teaching/sstutorial/
Park, Se Rim, and Jinwon Lee. arXiv preprint arXiv:1609.07132 (2016).

Introduction

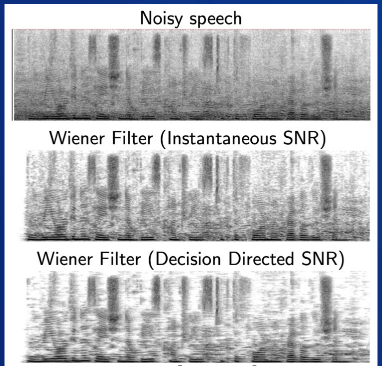
Audio Denoising is Hard for Wearable IoT Devices



$$|\hat{X}_m(\omega)|^2 = |Y_m(\omega)|^2 - E\left[|D_m(\omega)|^2\right]$$



$$|\hat{X}_m(\omega)| = |Y_m(\omega)| - E\left[|D_m(\omega)|\right]$$



$$\hat{X}(\omega) = \frac{E\left[|X(\omega)|^2\right]}{E[|X(\omega)|^2]+E[|D(\omega)|^2]} Y(\omega)$$

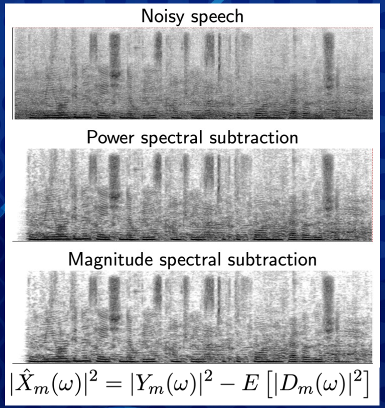
- Classical methods are rigid – noise estimation
  - *Fixed algorithms and parameters*

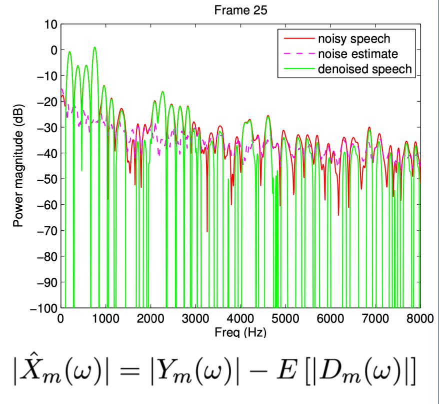
- Wearables require:
  - Superior audio quality
  - Low power consumption
  - Realtime performance

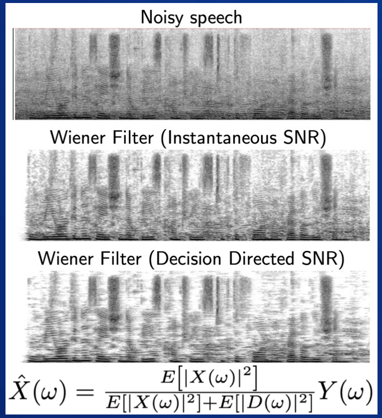
https://ccrma.stanford.edu/~njb/teaching/sstutorial/
Park, Se Rim, and Jinwon Lee. arXiv preprint arXiv:1609.07132 (2016).

Introduction

Audio Denoising is Hard for Wearable IoT Devices

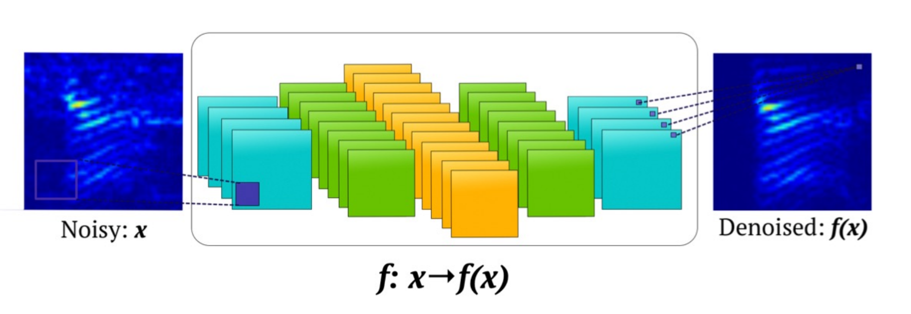






- Wearables require:
  - Superior audio quality
  - Low power consumption
  - Realtime performance

- Classical methods are rigid – noise estimation
  - *Fixed algorithms and parameters*



- CNNs offer flexibility but demand efficiency
  - *Generalizable across noise types*
  - *Finetune/retrain, downstream deploy*

https://ccrma.stanford.edu/~njb/teaching/sstutorial/
Park, Se Rim, and Jinwon Lee. arXiv preprint arXiv:1609.07132 (2016).

Introduction

Past ML-Based Audio Processing

Introduction

Past ML-Based Audio Processing

- **High Performance:**
  - Recent deep learning algorithms excel in audio processing

Introduction

Past ML-Based Audio Processing

- **High Performance:**
  - Recent deep learning algorithms excel in audio processing

- **Challenges:**
  - High computational complexity
  - Large model sizes
  - Substantial power and resource requirements

Introduction

Past ML-Based Audio Processing

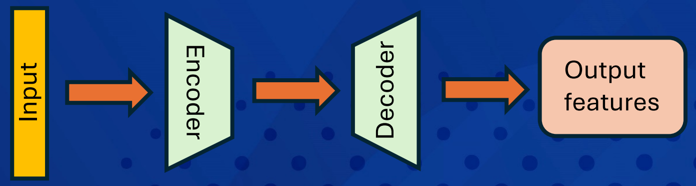
- **High Performance:**
  - Recent deep learning algorithms excel in audio processing
- **Challenges:**
  - High computational complexity
  - Large model sizes
  - Substantial power and resource requirements
- **Feasibility Issues:**
  - Unsuitable for IoT devices due to energy and size constraints

Introduction

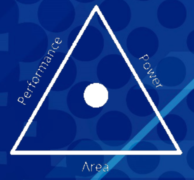
Past ML-Based Audio Processing

- **High Performance:**
  - Recent deep learning algorithms excel in audio processing
- **Challenges:**
  - High computational complexity
  - Large model sizes
  - Substantial power and resource requirements
- **Feasibility Issues:**
  - Unsuitable for IoT devices due to energy and size constraints

- Convolutional Encoder-Decoder (CED) models show promise in frequency-domain audio processing



- Practicality depends on **efficient hardware design** to reduce computational demands

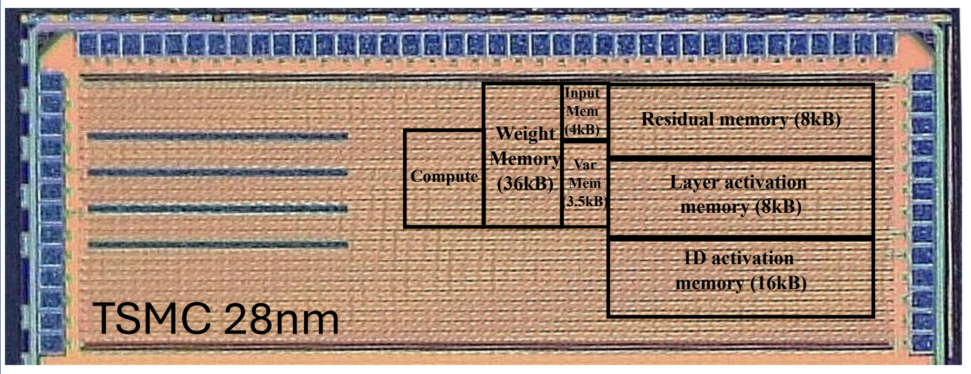


Introduction

Our Solution: A Real-Time Low-Power Denoising System

Introduction

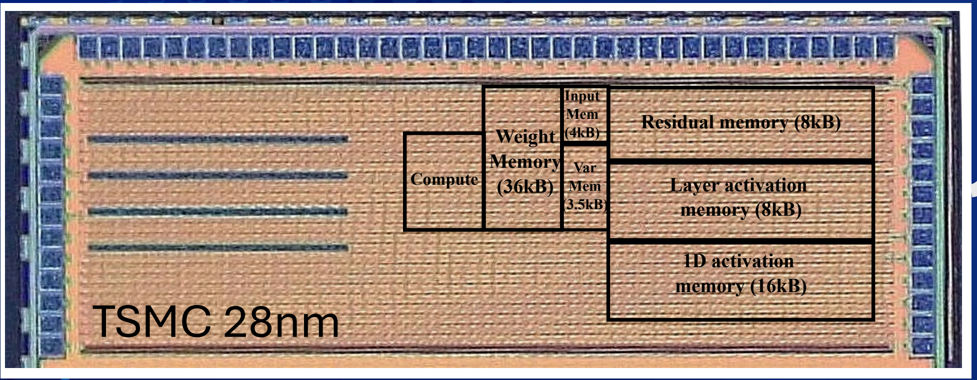
Our Solution: A Real-Time Low-Power Denoising System



Introduction

Our Solution: A Real-Time Low-Power Denoising System

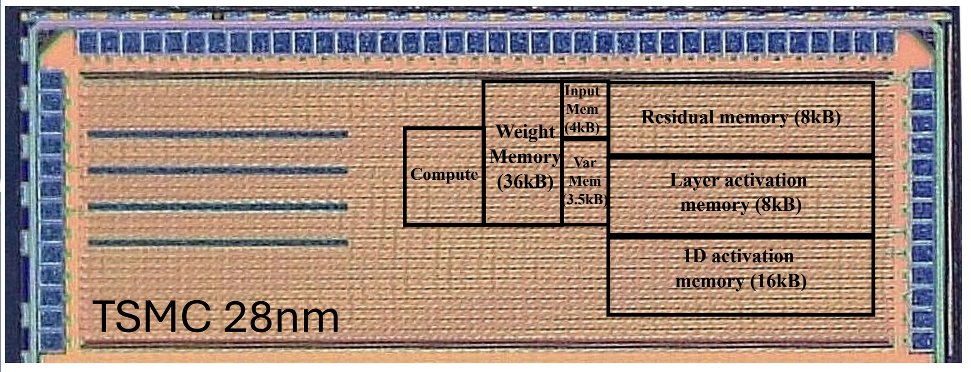
Lower computational costs with optimized quantization



Introduction

Our Solution: A Real-Time Low-Power Denoising System

Lower computational costs
with optimized quantization

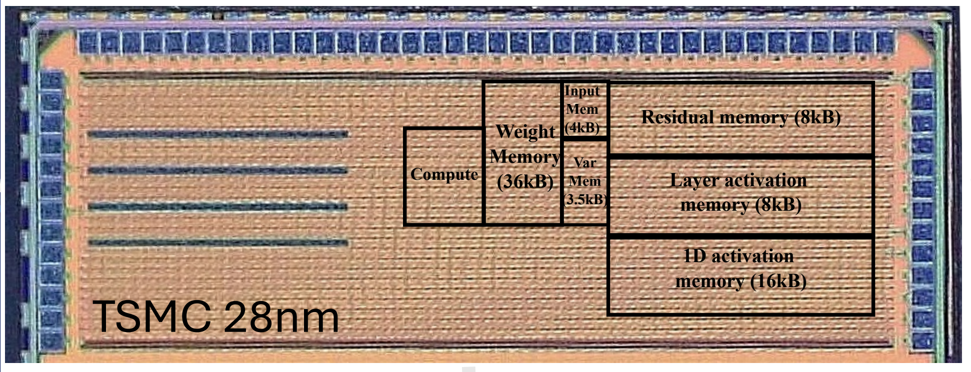


Low on-chip
memory accesses,
highest audio
quality score

Introduction

Our Solution: A Real-Time Low-Power Denoising System

Lower computational costs with optimized quantization



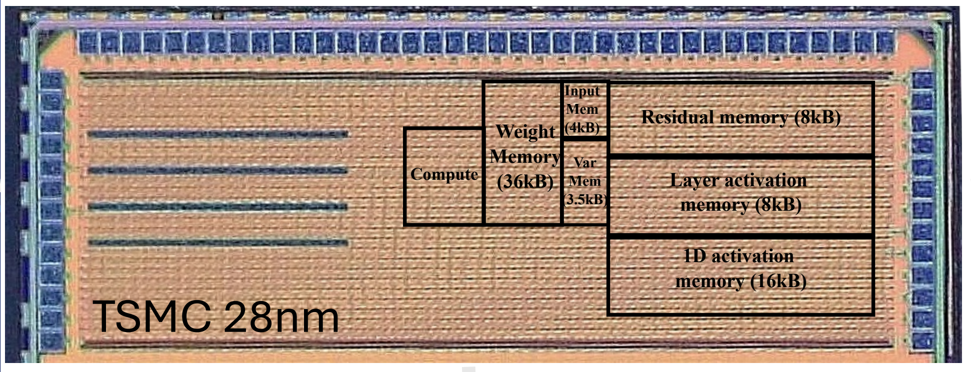
Low on-chip memory accesses, highest audio quality score

Processes audio in 8ms per frame, consumes 407µW

Introduction

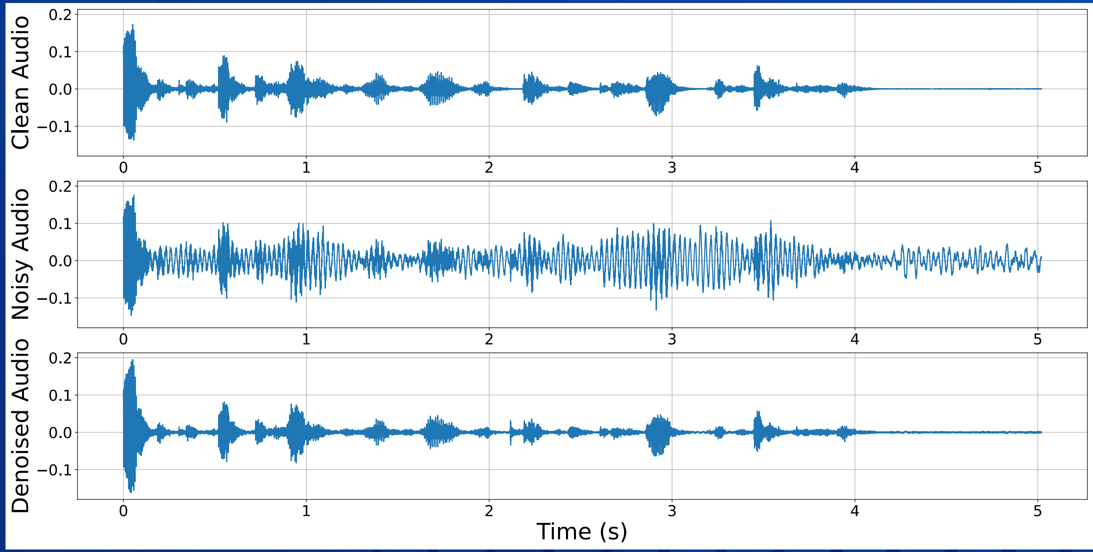
Our Solution: A Real-Time Low-Power Denoising System

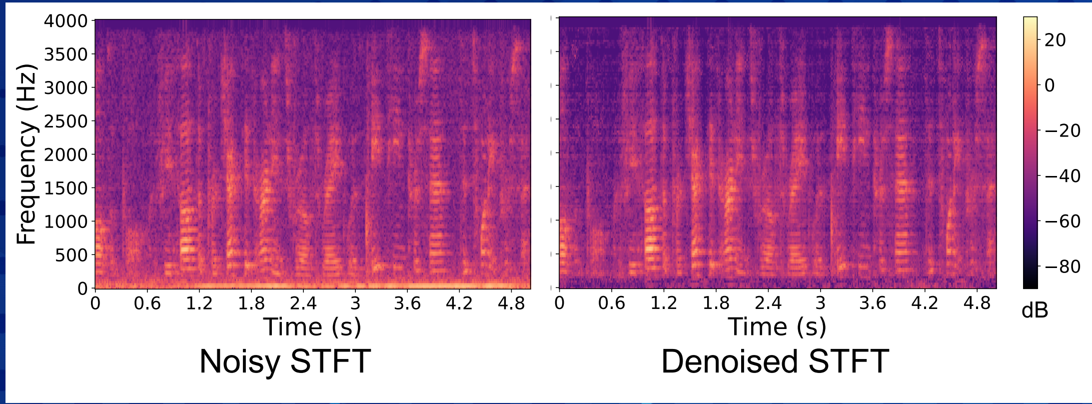
Lower computational costs with optimized quantization



Low on-chip memory accesses, highest audio quality score

Processes audio in 8ms per frame, consumes 407µW





Outline

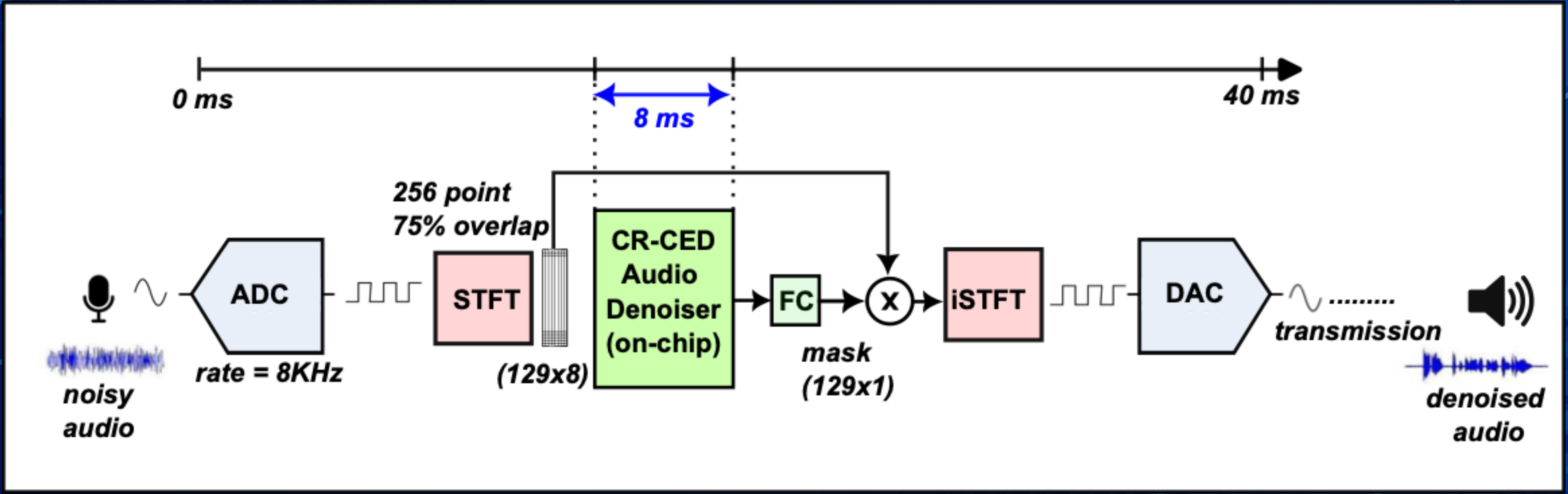
- Introduction
- **Design Features**
  - **Algorithm Design**
  - Quantization Scheme
  - Top-level Chip Architecture
  - 1D Convolution Dataflow
- Results
- Conclusion

Algorithm Design

End-to-end Audio Denoiser Pipeline

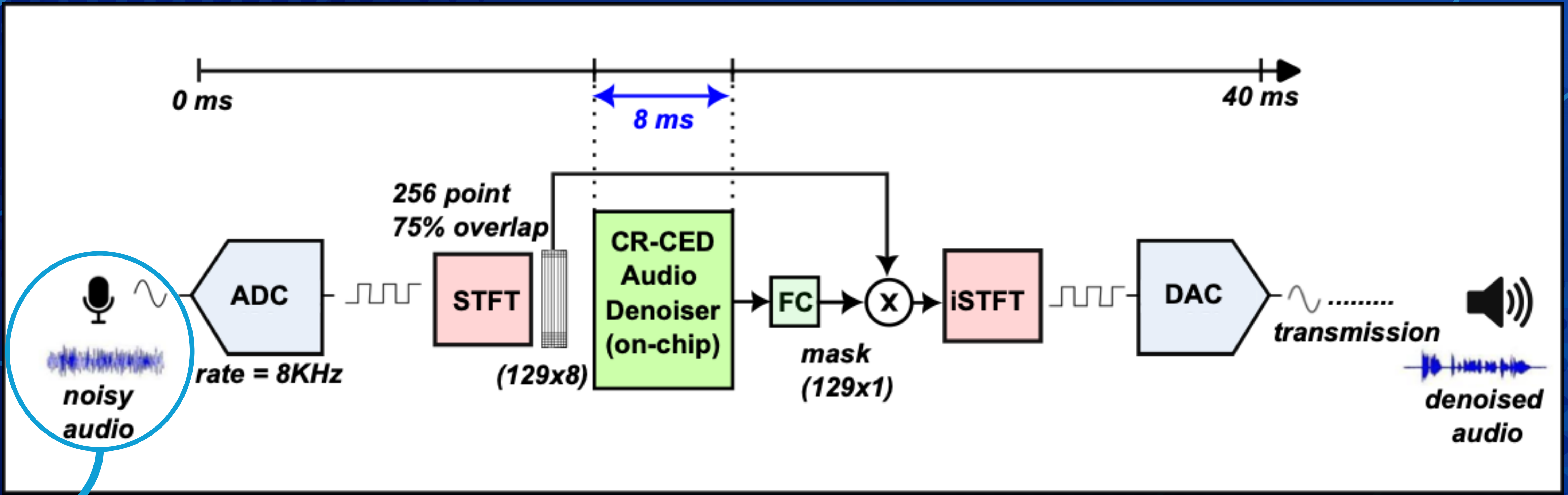
Algorithm Design

End-to-end Audio Denoiser Pipeline



Algorithm Design

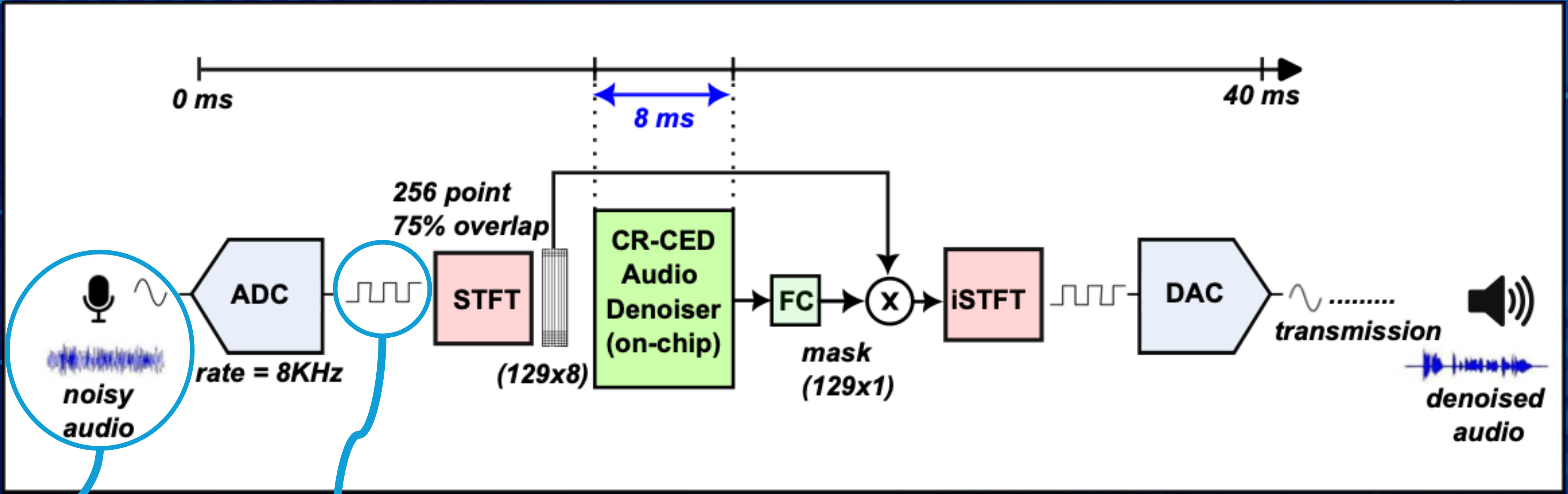
End-to-end Audio Denoiser Pipeline



Audio captured by microphone

Algorithm Design

End-to-end Audio Denoiser Pipeline

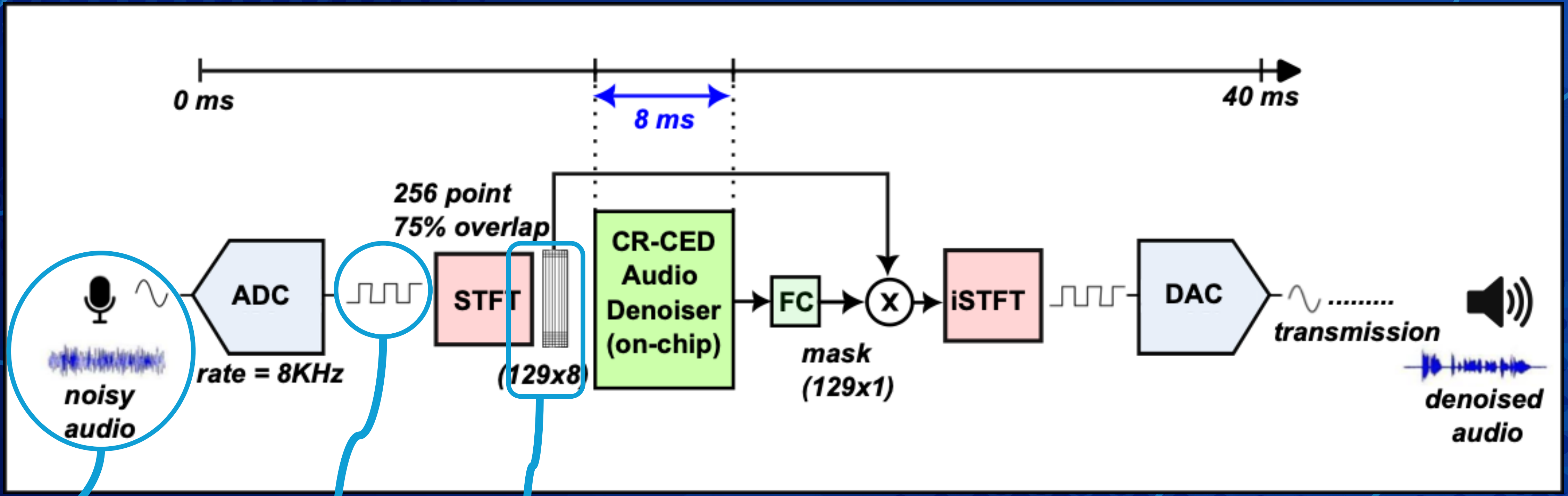


Audio captured by microphone

1D Time series

Algorithm Design

End-to-end Audio Denoiser Pipeline



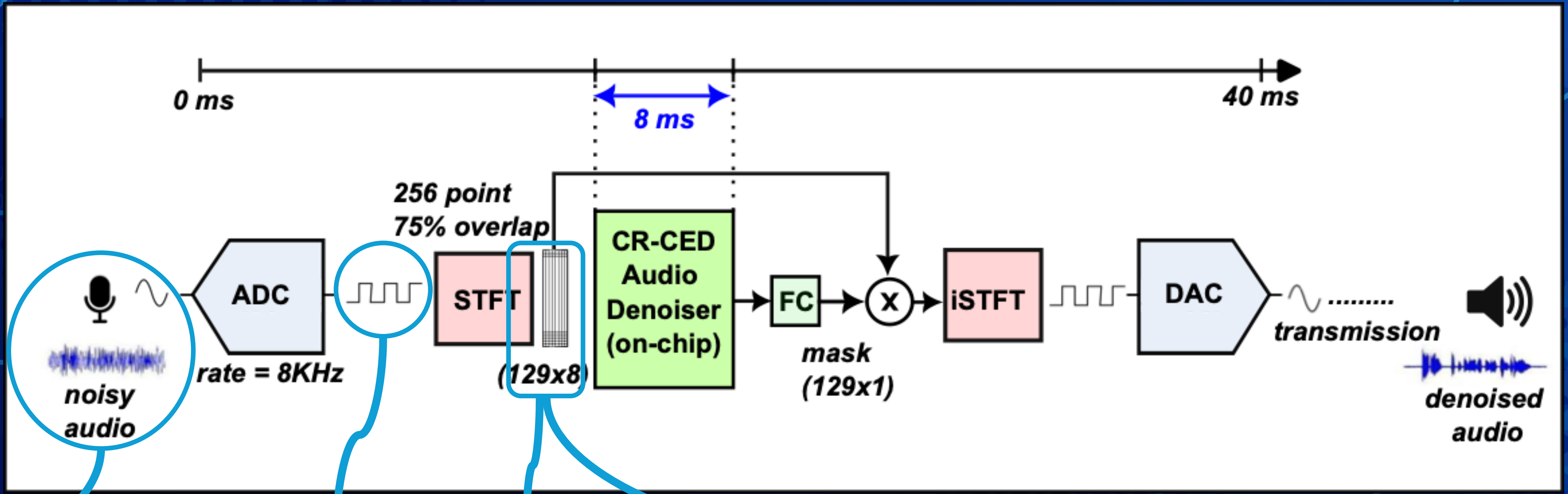
Audio captured by microphone

1D Time series

2D Time – Frequency series

Algorithm Design

End-to-end Audio Denoiser Pipeline



Audio captured by microphone

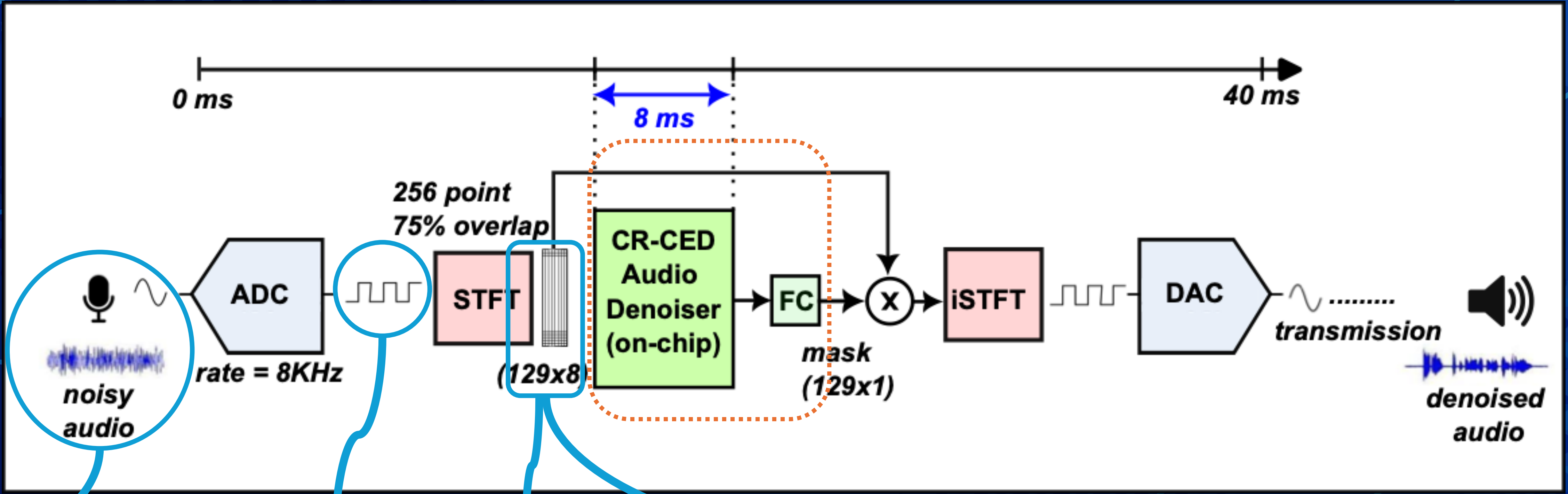
1D Time series

2D Time – Frequency series

129: Magnitude vectors from the 256-point STFT, 129 points retained (symmetric half of 256)
8: Concatenate eight consecutive STFT vectors: past five, current, and future two noise contexts

Algorithm Design

End-to-end Audio Denoiser Pipeline



Audio captured by microphone

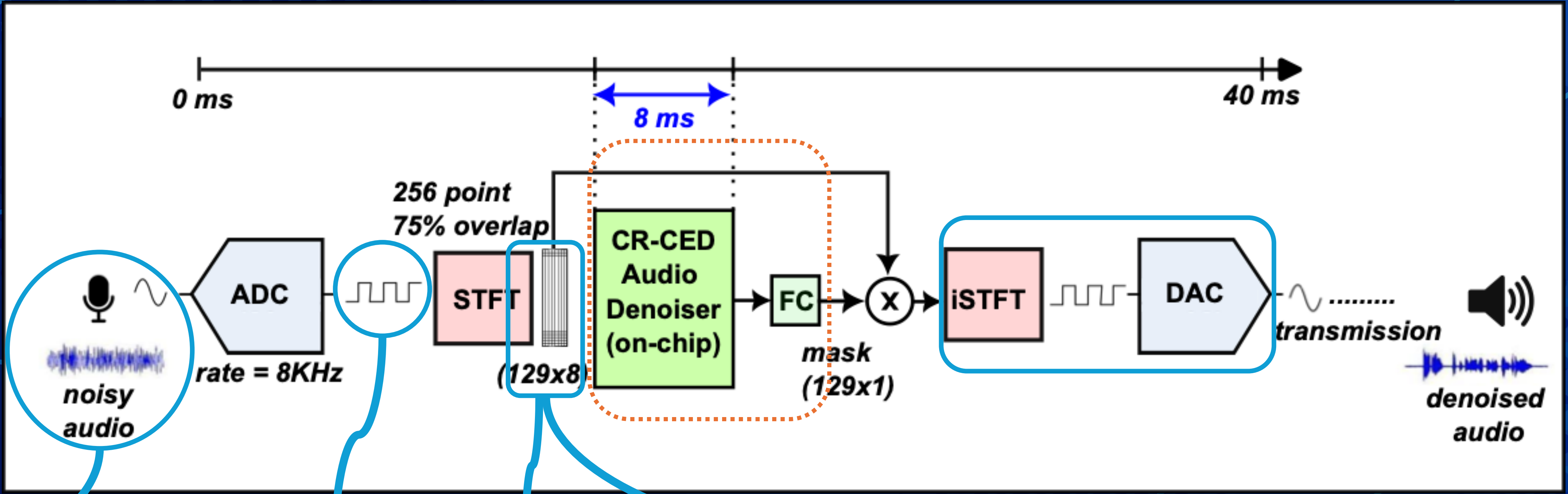
1D Time series

2D Time – Frequency series

129: Magnitude vectors from the 256-point STFT, 129 points retained (symmetric half of 256)
8: Concatenate eight consecutive STFT vectors: past five, current, and future two noise contexts

Algorithm Design

End-to-end Audio Denoiser Pipeline



Audio captured by microphone

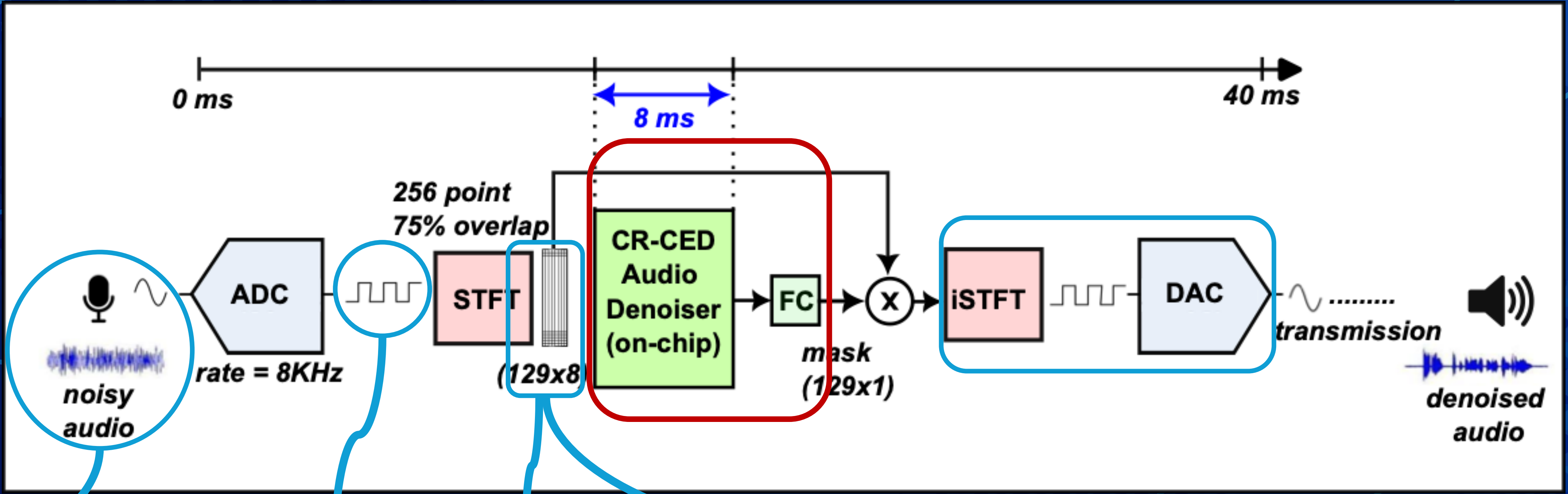
1D Time series

2D Time – Frequency series

129: Magnitude vectors from the 256-point STFT, 129 points retained (symmetric half of 256)
8: Concatenate eight consecutive STFT vectors: past five, current, and future two noise contexts

Algorithm Design

End-to-end Audio Denoiser Pipeline



Audio captured by microphone

1D Time series

2D Time – Frequency series

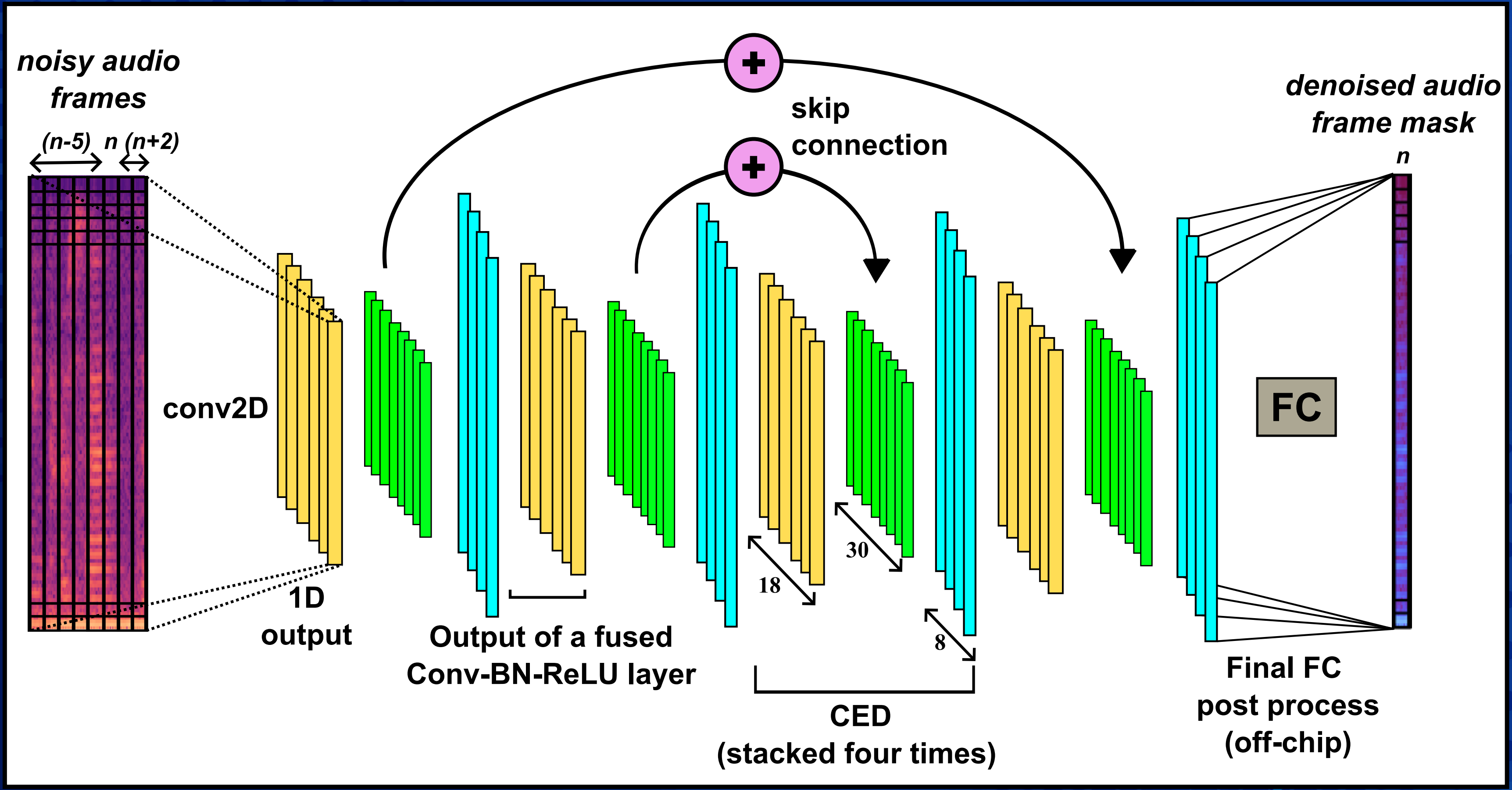
129: Magnitude vectors from the 256-point STFT, 129 points retained (symmetric half of 256)
8: Concatenate eight consecutive STFT vectors: past five, current, and future two noise contexts

Algorithm Design

Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)

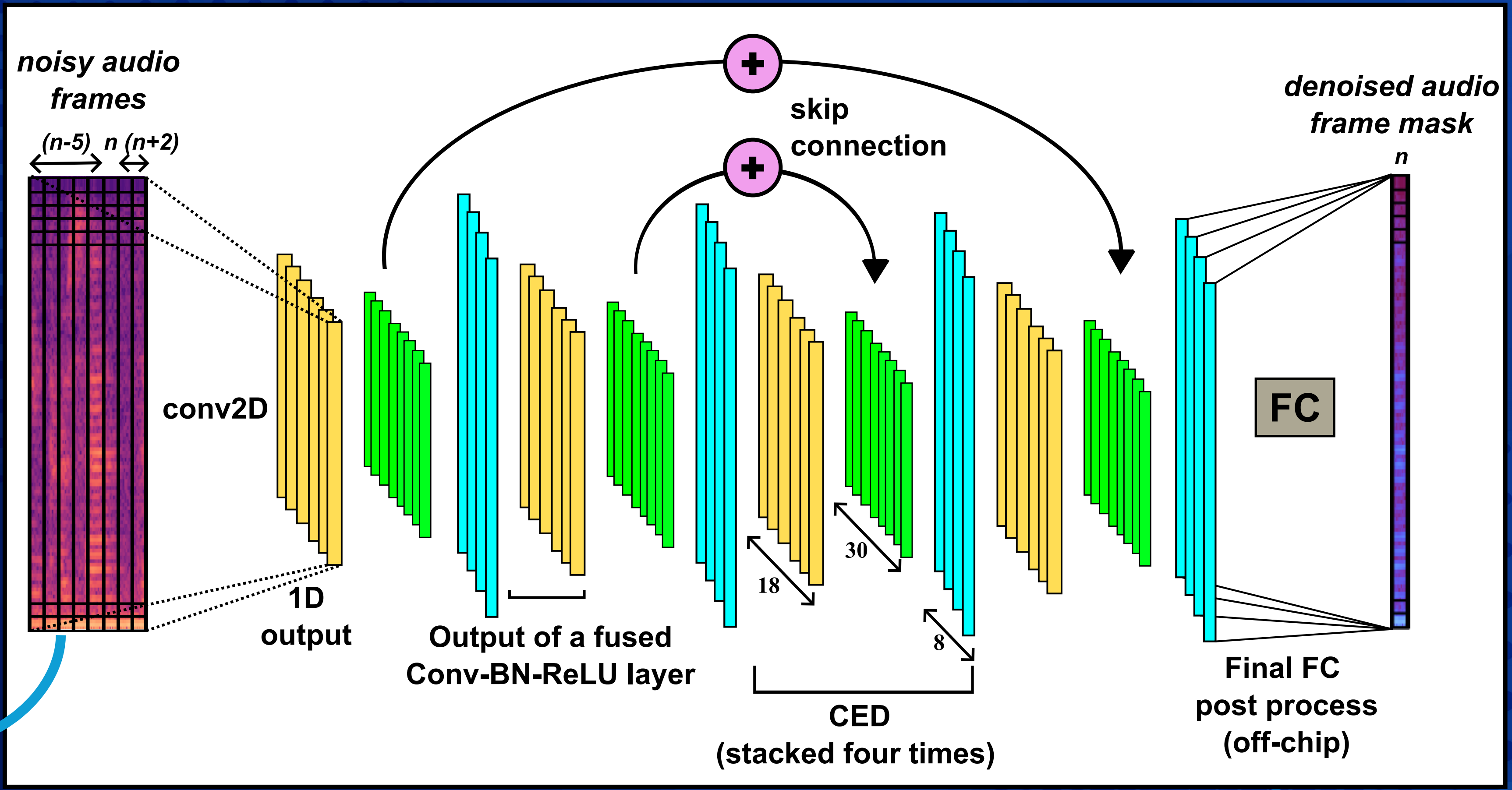
Algorithm Design

Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)



Algorithm Design

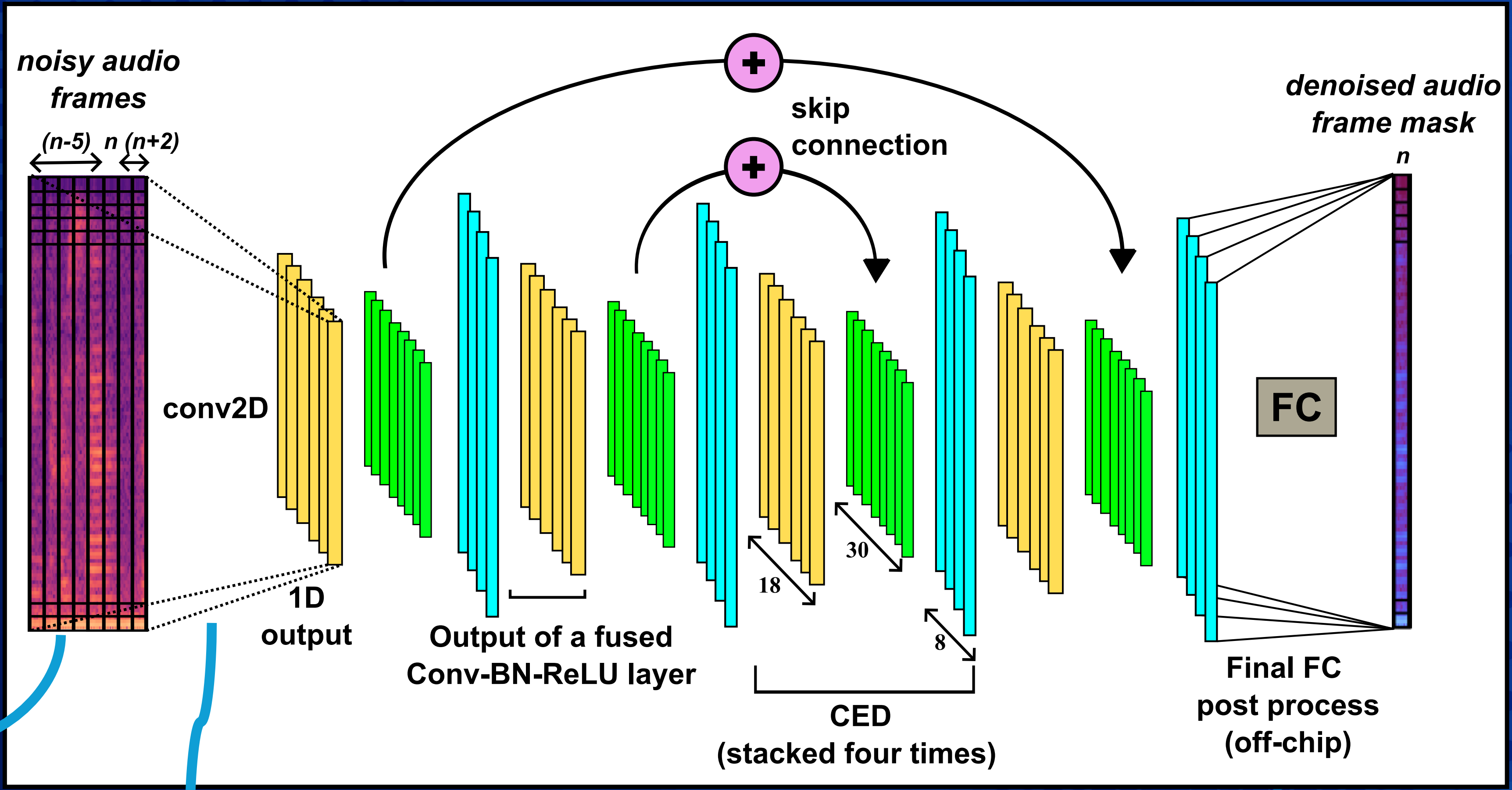
Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)



Treated as 2D image

Algorithm Design

Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)

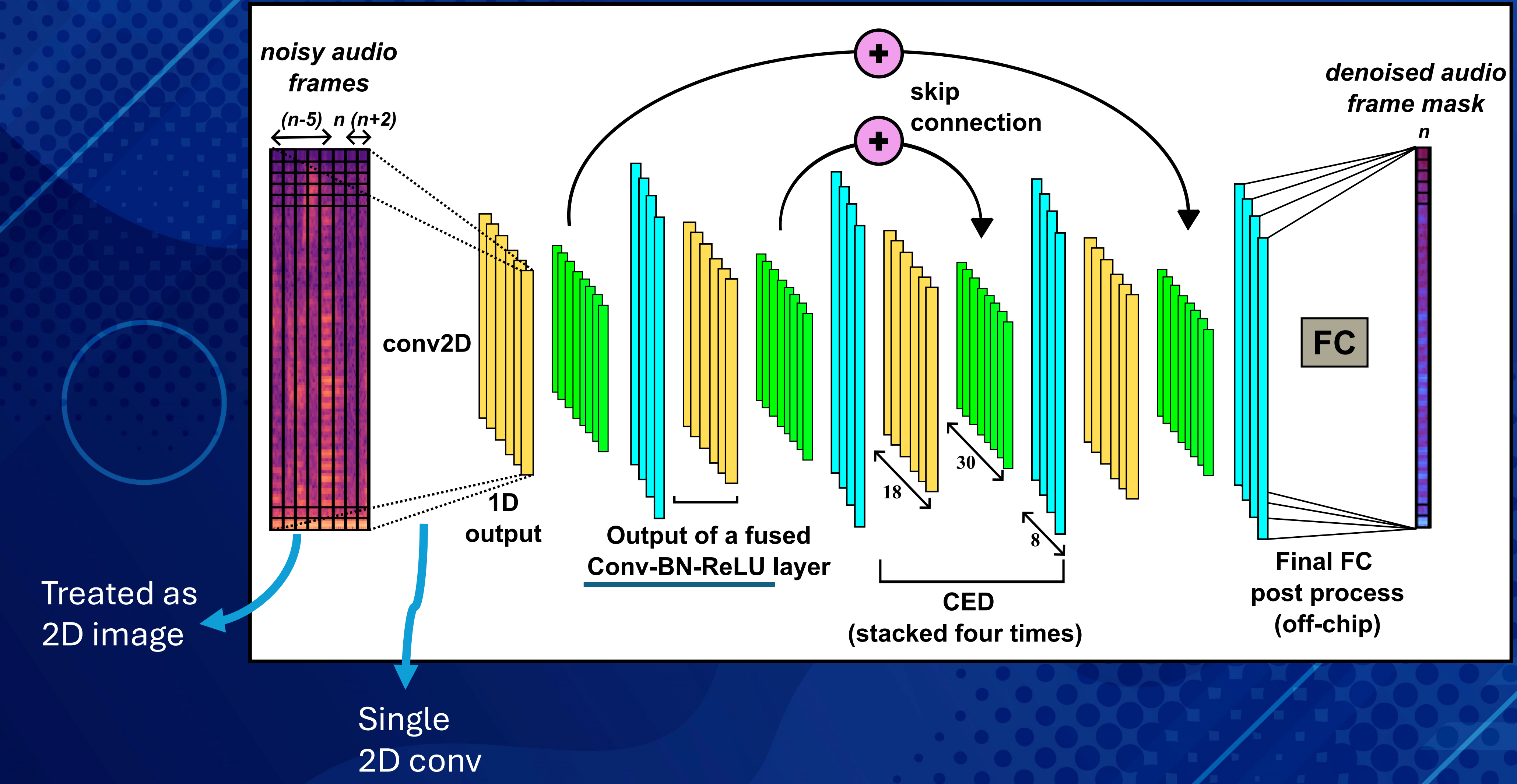


Treated as 2D image

Single 2D conv

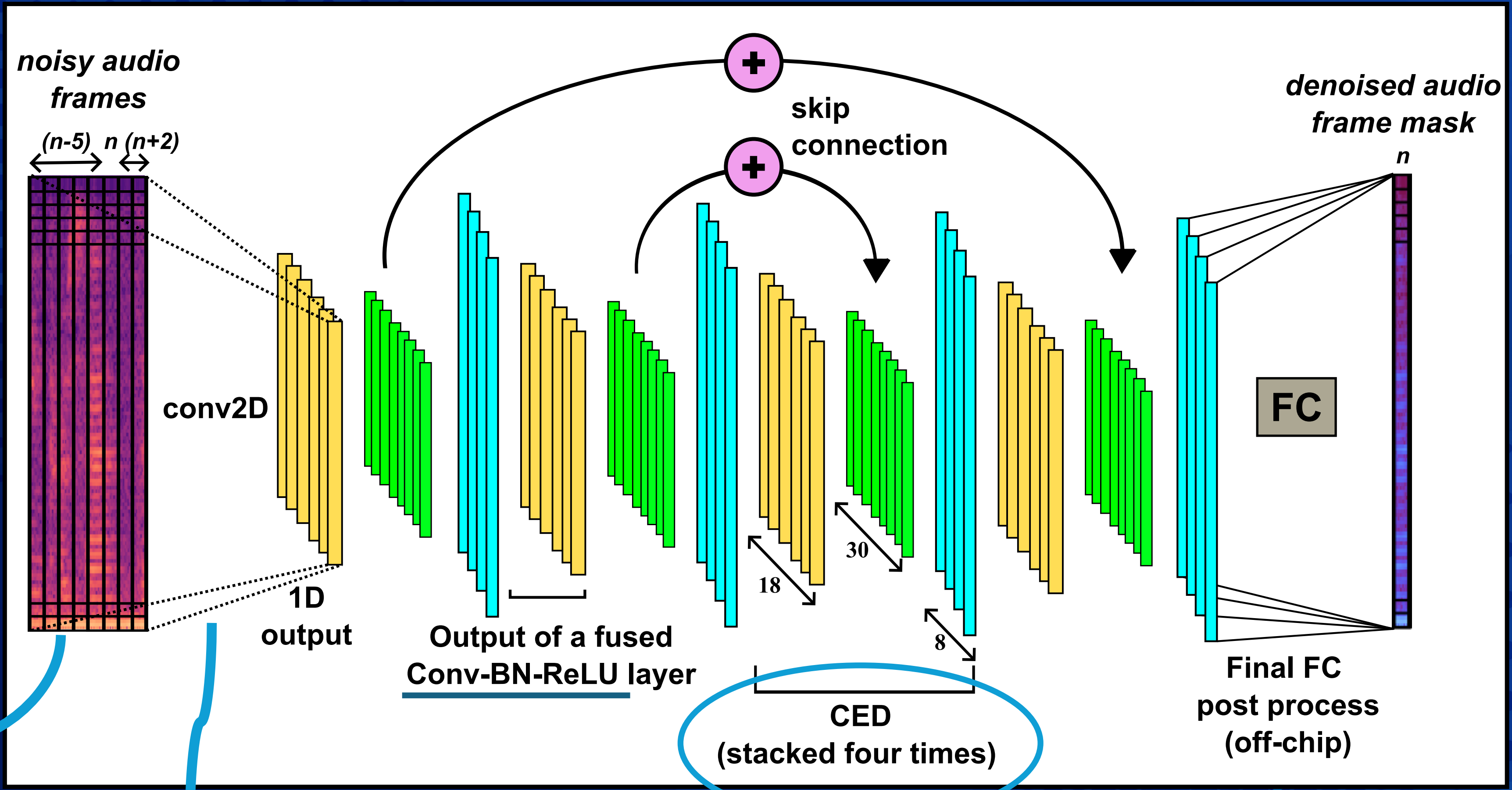
Algorithm Design

Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)



Algorithm Design

Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)

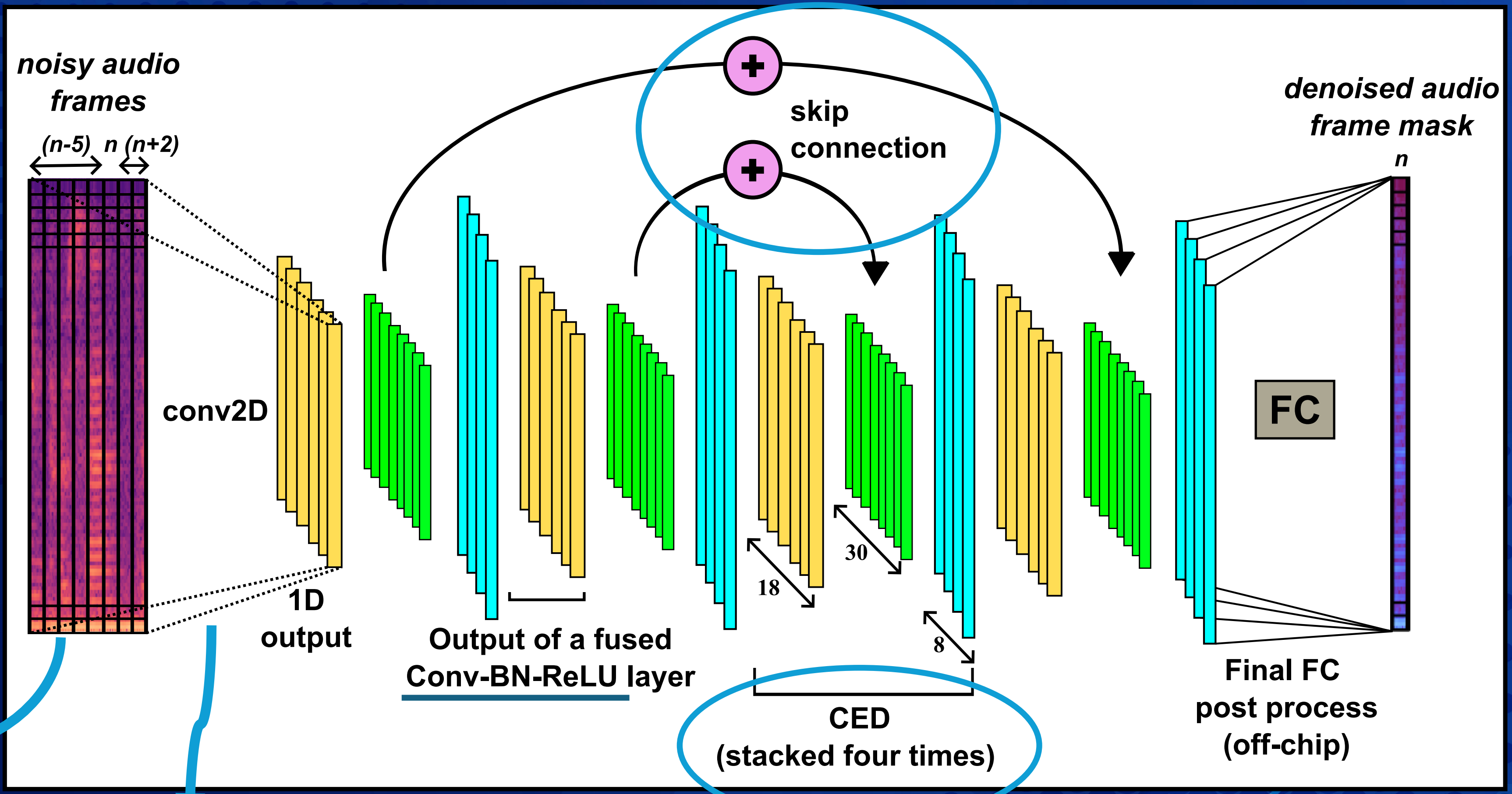


Treated as 2D image

Single 2D conv

Algorithm Design

Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)

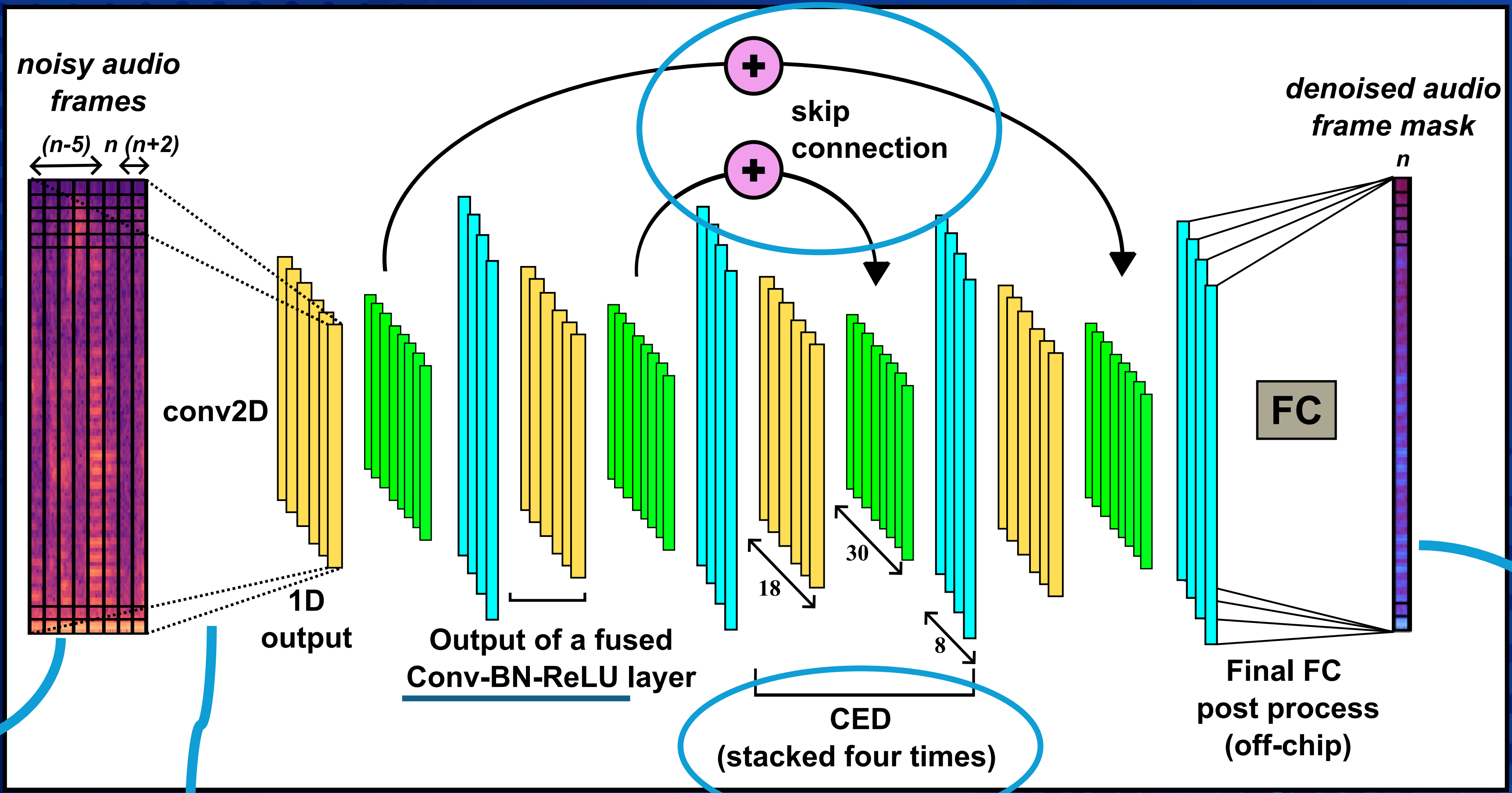


Treated as 2D image

Single 2D conv

Algorithm Design

Neural Network: Cascaded Redundant Convolutional Encoder-Decoder (CR-CED)



Treated as 2D image

Single 2D conv

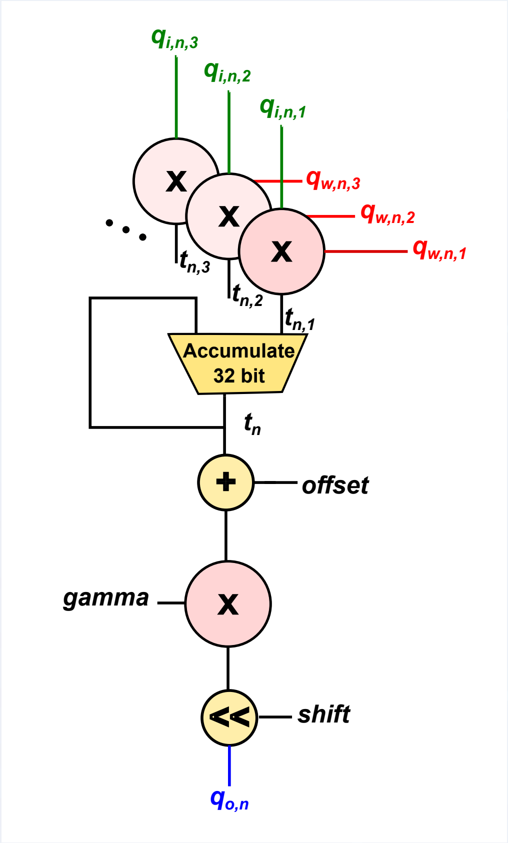
Mask vector: multiplied by the noisy frame to obtain the denoised audio STFT

Outline

- Introduction
- **Design Features**
  - Algorithm Design
  - **Quantization Scheme**
  - Top-level Chip Architecture
  - 1D Convolution Dataflow
- Results
- Conclusion

Quantization Scheme

8-bit weight, activation quantization



Quantization Scheme

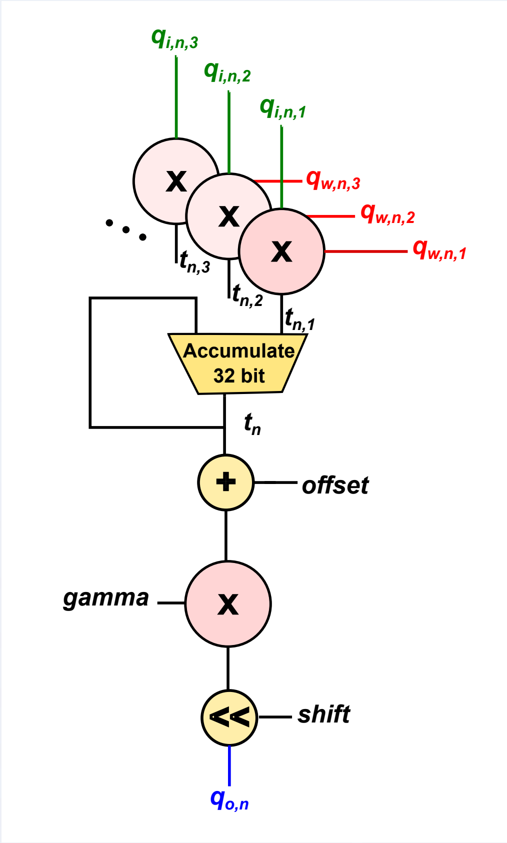
8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w);.\ z_w = 0$

Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

Offset: $b = s_b * (q_b - z_b);.\ z_b = 0$



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = \boxed{s_w} * (q_w - z_w);.\ z_w = 0$

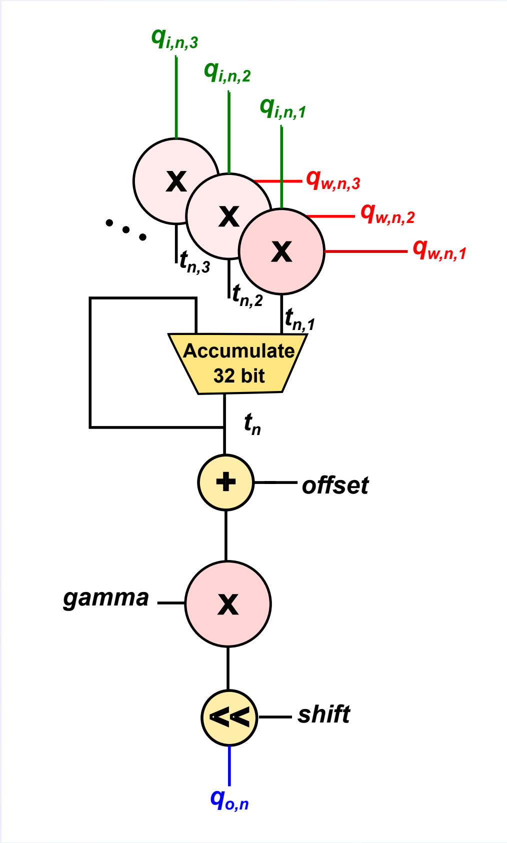
Input: $i = \boxed{s_i} * (q_i - z_i)$

Output: $o = \boxed{s_o} * (q_o - z_o)$

Offset: $b = \boxed{s_b} * (q_b - z_b);.\ z_b = 0$

float32 scale factors:

s_w, s_i, s_o, s_b



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w); . \ z_w = 0$

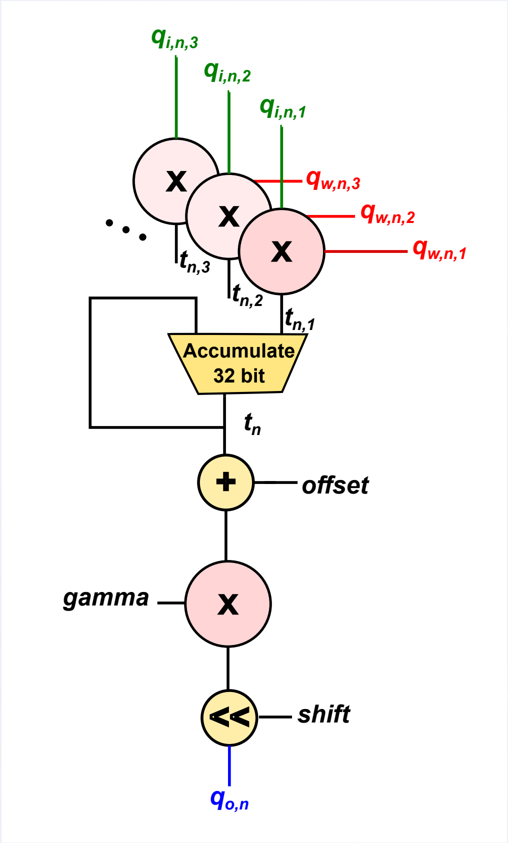
Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

Offset: $b = s_b * (q_b - z_b); . \ z_b = 0$

float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w);.\ z_w = 0$

Input: $i = s_i * (q_i - z_i)$

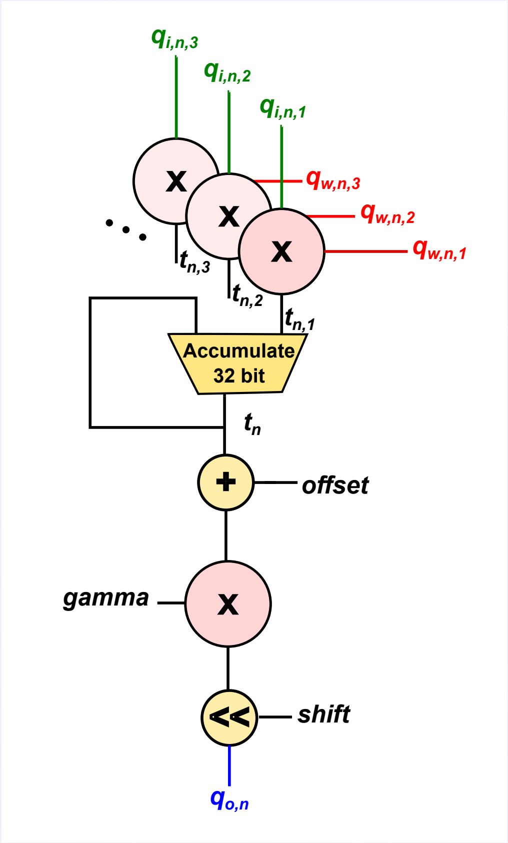
Output: $o = s_o * (q_o - z_o)$

Offset: $b = s_b * (q_b - z_b);.\ z_b = 0$

float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b

8-bit zero-points: z_w, z_i, z_o, z_b



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w);.\ z_w = 0$

Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

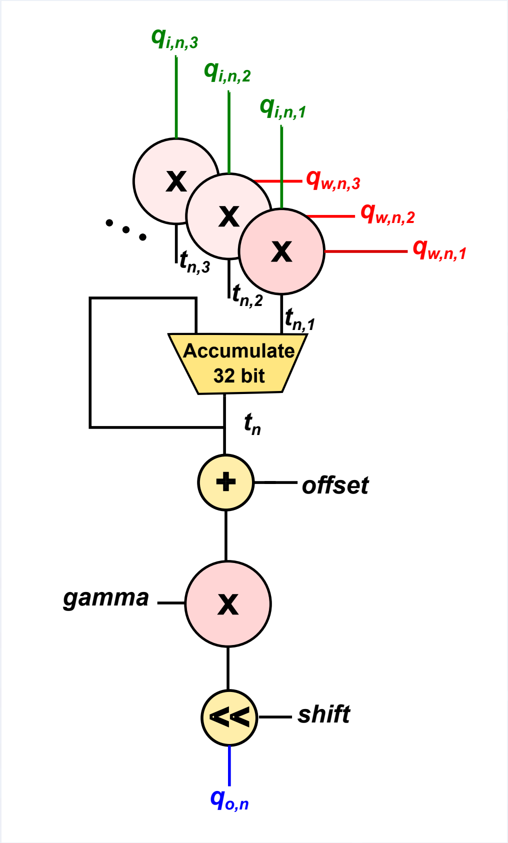
Offset: $b = s_b * (q_b - z_b);.\ z_b = 0$

float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b

8-bit zero-points: z_w, z_i, z_o, z_b

$$o_n = \Sigma\, w_{n,k} * i_{n,k} + b_n$$



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w);.\ z_w = 0$

Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

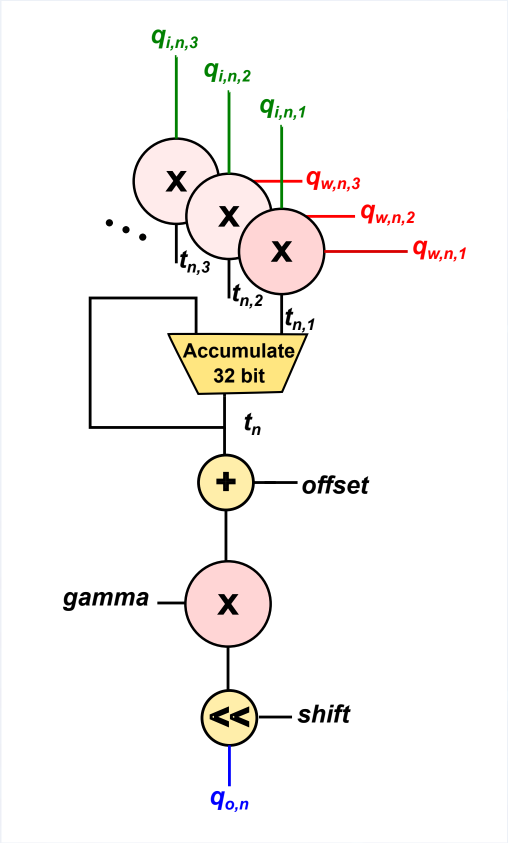
Offset: $b = s_b * (q_b - z_b);.\ z_b = 0$

float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b

8-bit zero-points: z_w, z_i, z_o, z_b

$$o_n = \Sigma\, w_{n,k} * i_{n,k} + b_n$$



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w);\ .\ z_w = 0$

Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

Offset: $b = s_b * (q_b - z_b);\ .\ z_b = 0$

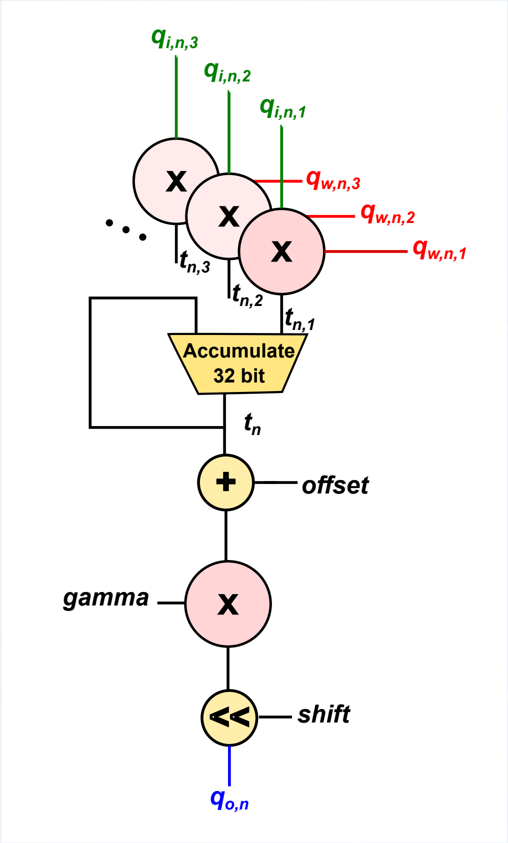
float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b

8-bit zero-points: z_w, z_i, z_o, z_b

$$o_n = \Sigma\, w_{n,k} * i_{n,k} + b_n$$

$$s_{o,n} * (q_{o,n} - z_{o,n}) = \Sigma\,[s_{w,n} * q_{w,n,k} * s_{i,n} * (q_{i,n,k} - z_{i,n})] + s_{b,n} * q_{b,n}$$



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w);.\ z_w = 0$

Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

Offset: $b = s_b * (q_b - z_b);.\ z_b = 0$

float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b

8-bit zero-points: z_w, z_i, z_o, z_b

$$o_n = \Sigma\, w_{n,k} * i_{n,k} + b_n$$

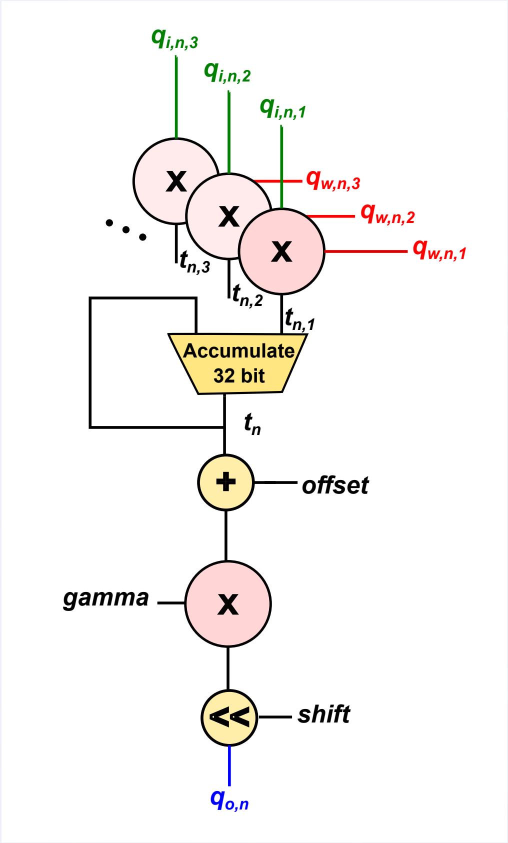
$$s_{o,n} * (q_{o,n} - z_{o,n}) = \Sigma\, [s_{w,n} * q_{w,n,k} * s_{i,n} * (q_{i,n,k} - z_{i,n})] + s_{b,n} * q_{b,n}$$

$$q_{o,n} = [gamma_n * (\Sigma\, q_{w,n,k} * q_{i,n,k} + offset_n)] \gg shift_n$$

Where,

$$gamma_n \gg shift_n = s_{w,n} * s_{i,n} * s_{o,n}^{-1}$$

$$offset_n = (z_{o,n} s_{o,n} + s_{b,n} q_{b,n}) s_{w,n}^{-1} s_{i,n}^{-1} - z_{i,n} \Sigma\, q_{w,n,k}$$



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w);.\ z_w = 0$

Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

Offset: $b = s_b * (q_b - z_b);.\ z_b = 0$

float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b

8-bit zero-points: z_w, z_i, z_o, z_b

$$o_n = \Sigma\, w_{n,k} * i_{n,k} + b_n$$

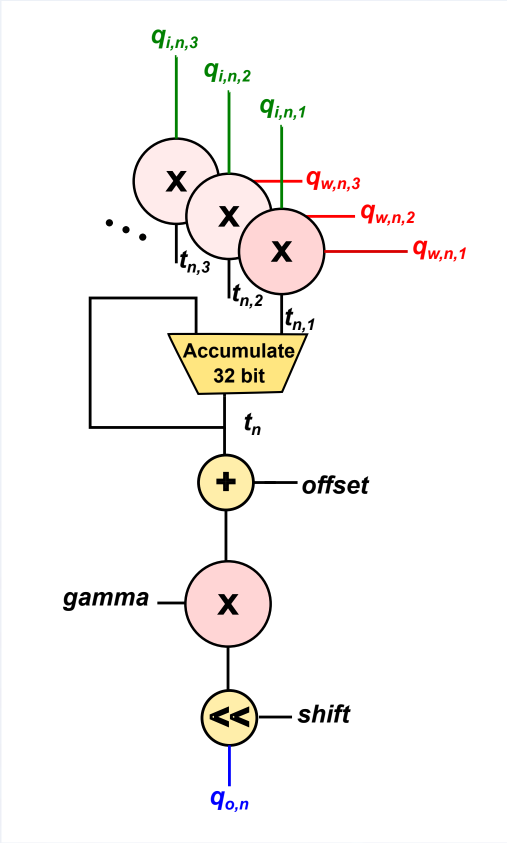
$$s_{o,n} * (q_{o,n} - z_{o,n}) = \Sigma\, [s_{w,n} * q_{w,n,k} * s_{i,n} * (q_{i,n,k} - z_{i,n})] + s_{b,n} * q_{b,n}$$

$$q_{o,n} = [gamma_n * (\Sigma\, q_{w,n,k} * q_{i,n,k} + offset_n\ )] \gg shift_n$$

Where,

$$gamma_n \gg shift_n = s_{w,n} * s_{i,n} * s_{o,n}^{-1}$$

$$offset_n = (z_{o,n} s_{o,n} + s_{b,n} q_{b,n}) s_{w,n}^{-1}\, s_{i,n}^{-1} - z_{i,n} \Sigma\, q_{w,n,k}$$



Quantization Scheme

8-bit weight, activation quantization

Weight: $w = s_w * (q_w - z_w)$; $z_w = 0$

Input: $i = s_i * (q_i - z_i)$

Output: $o = s_o * (q_o - z_o)$

Offset: $b = s_b * (q_b - z_b)$; $z_b = 0$

float32 scale factors: s_w, s_i, s_o, s_b

8-bit quantized values stored on-chip: q_w, q_i, q_o, q_b

8-bit zero-points: z_w, z_i, z_o, z_b

$$o_n = \Sigma\, w_{n,k} * i_{n,k} + b_n$$

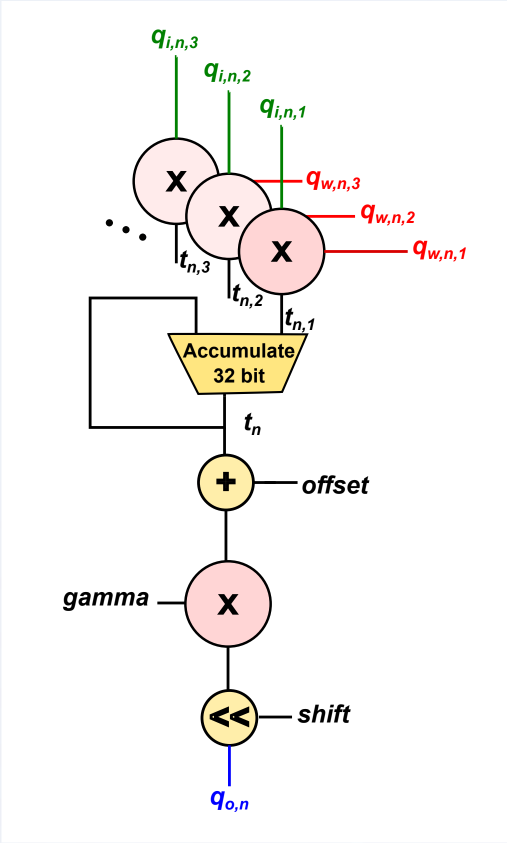
$$s_{o,n} * (q_{o,n} - z_{o,n}) = \Sigma\, [s_{w,n} * q_{w,n,k} * s_{i,n} * (q_{i,n,k} - z_{i,n})] + s_{b,n} * q_{b,n}$$

$$q_{o,n} = [gamma_n * (\Sigma\, \boxed{q_{w,n,k} * q_{i,n,k}} + offset_n\,)] \gg shift_n$$

Where,

$$gamma_n \gg shift_n = s_{w,n} * s_{i,n} * s_{o,n}^{-1}$$

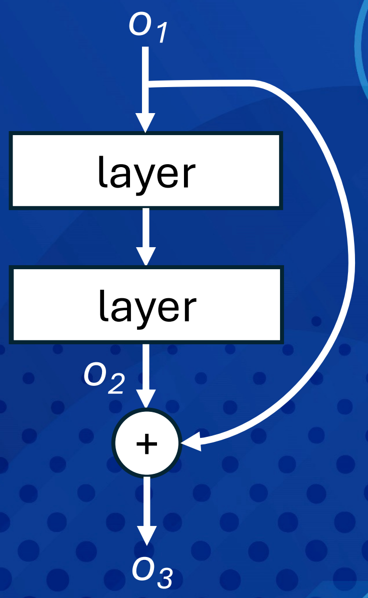
$$offset_n = (z_{o,n} s_{o,n} + s_{b,n} q_{b,n}) s_{w,n}^{-1}\, s_{i,n}^{-1} - z_{i,n} \Sigma\, q_{w,n,k}$$



Minimal drop in performance
2.83 to 2.79 PESQ
in the audio quality evaluation score

Quantization Scheme

Skip Connections, Per-kernel Adaptive Rounding



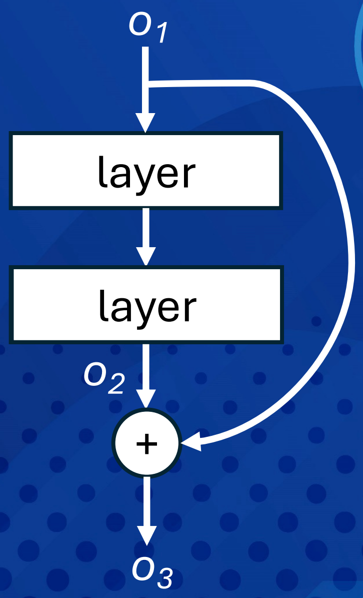
Quantization Scheme

Skip Connections, Per-kernel Adaptive Rounding

Skip connection computation:

$$o_3 = o_1 + o_2$$

$$s_3(q_{o,3} - z_3) = s_1(q_{o,1} - z_1) + s_2(q_{o,2} - z_2)$$



Quantization Scheme

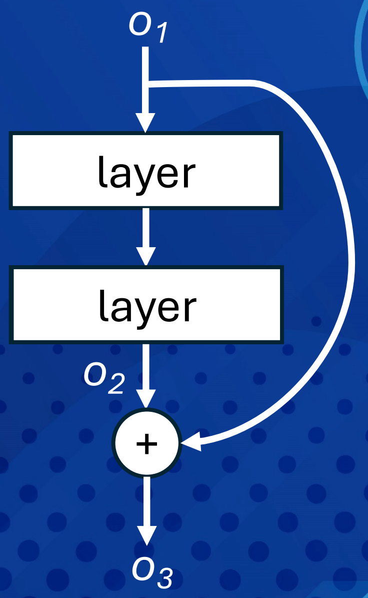
Skip Connections, Per-kernel Adaptive Rounding

Skip connection computation:

$$o_3 = o_1 + o_2$$

$$s_3(q_{o,3} - z_3) = s_1(q_{o,1} - z_1) + s_2(q_{o,2} - z_2)$$

$$q_{o,3} = (s_{skip,1} q_{o,1} + s_{skip,2} q_{o,2} + offset_{skip}) \gg shift_{skip}$$



Quantization Scheme

Skip Connections, Per-kernel Adaptive Rounding

Skip connection computation:

$$o_3 = o_1 + o_2$$

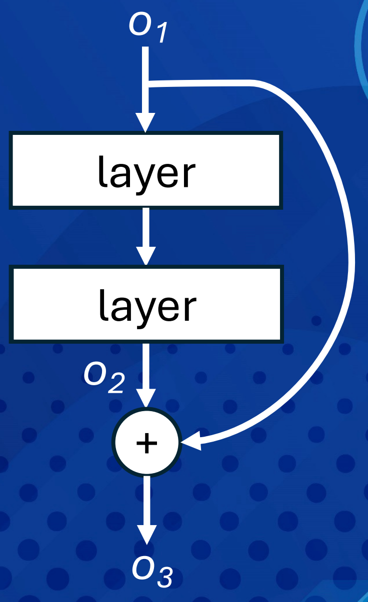
$$s_3(q_{o,3} - z_3) = s_1(q_{o,1} - z_1) + s_2(q_{o,2} - z_2)$$

$$q_{o,3} = (s_{skip,1}q_{o,1} + s_{skip,2}q_{o,2} + offset_{skip}) \gg shift_{skip}$$

Where,

$$s_{skip,i} \gg shift_{skip} = \frac{s_i}{s_3}, \qquad i = 1, 2$$

$$offset_{skip} \gg shift_{skip} = z_3 - \left(\frac{s_1}{s_3}\right) z_1 - \left(\frac{s_2}{s_3}\right) z_2$$



Quantization Scheme

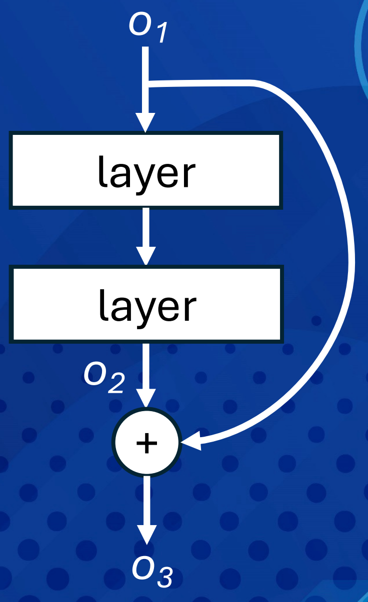
Skip Connections, Per-kernel Adaptive Rounding

Skip connection computation:

$$o_3 = o_1 + o_2$$

$$s_3(q_{o,3} - z_3) = s_1(q_{o,1} - z_1) + s_2(q_{o,2} - z_2)$$

$$q_{o,3} = (s_{skip,1} q_{o,1} + s_{skip,2} q_{o,2} + offset_{skip}) \gg shift_{skip}$$



Where,

$$s_{skip,i} \gg shift_{skip} = \frac{s_i}{s_3}, \qquad i = 1,2$$

$$offset_{skip} \gg shift_{skip} = z_3 - \left(\frac{s_1}{s_3}\right) z_1 - \left(\frac{s_2}{s_3}\right) z_2$$

Per-kernel adaptive rounding: Determines how to round and at which precision

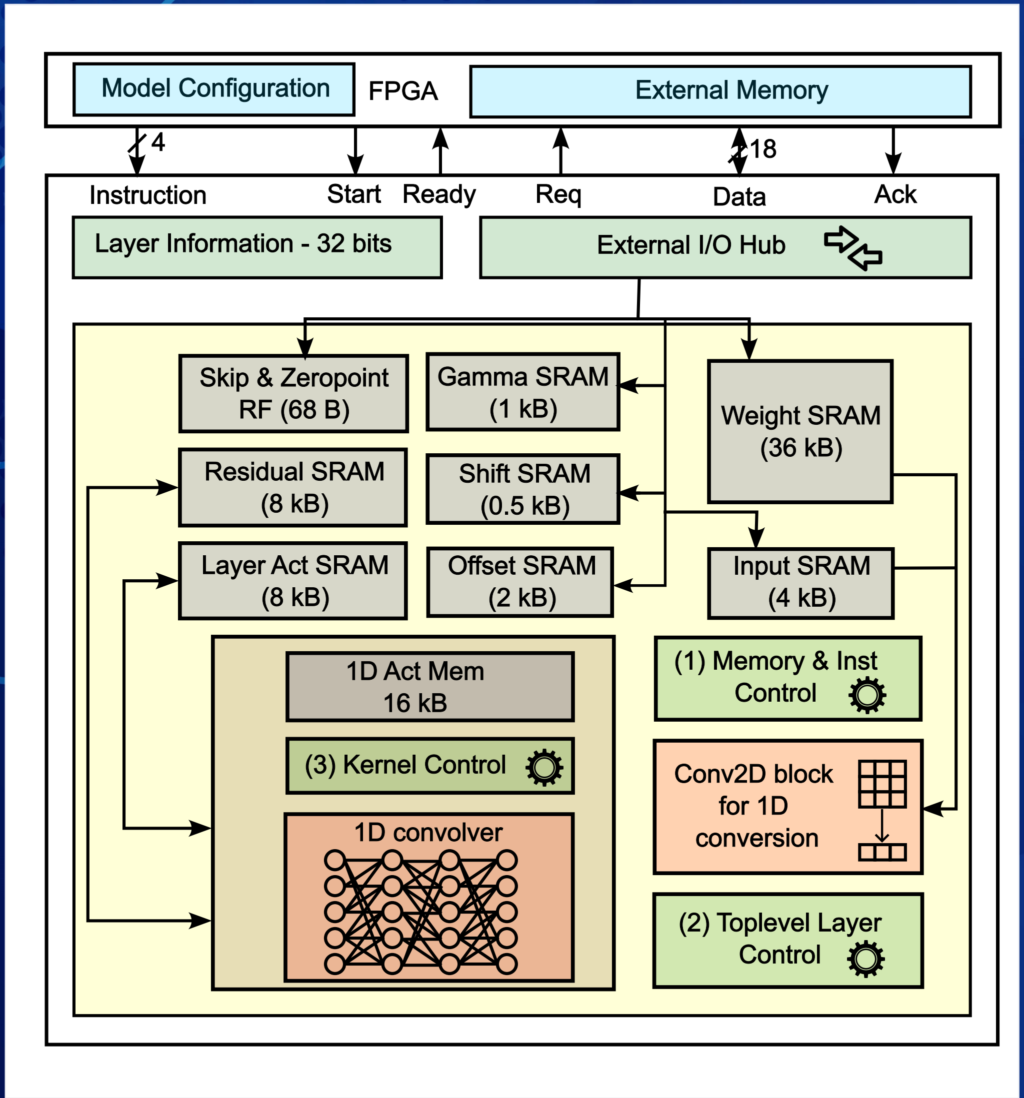
Adding a constant to *offset*$_n$ while eliminating the need for a comparator

Outline

- Introduction
- **Design Features**
  - Algorithm Design
  - Quantization Scheme
  - **Top-level Chip Architecture**
  - 1D Convolution Dataflow
- Results
- Conclusion

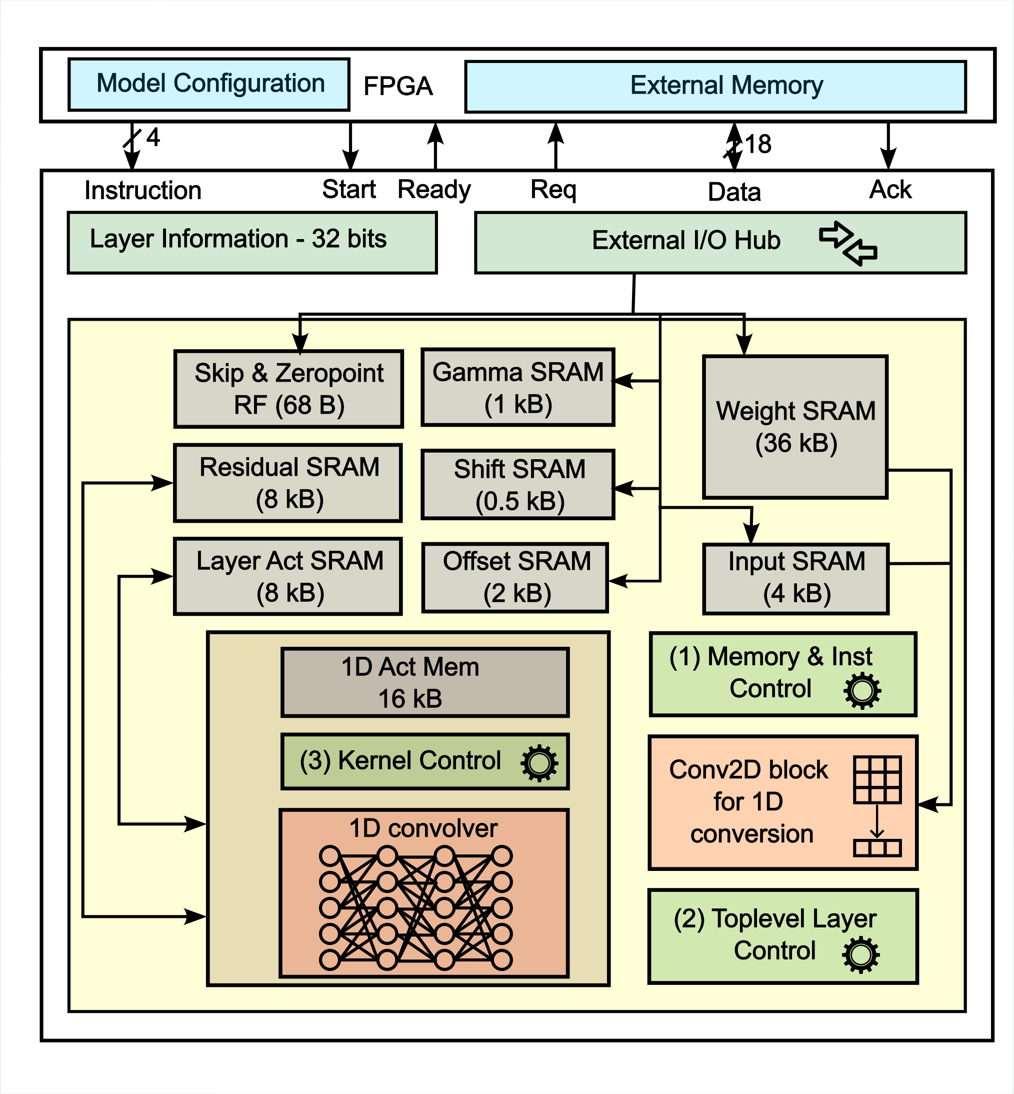
Top-level Chip Architecture

High Level Computation and Memory Blocks



Top-level Chip Architecture

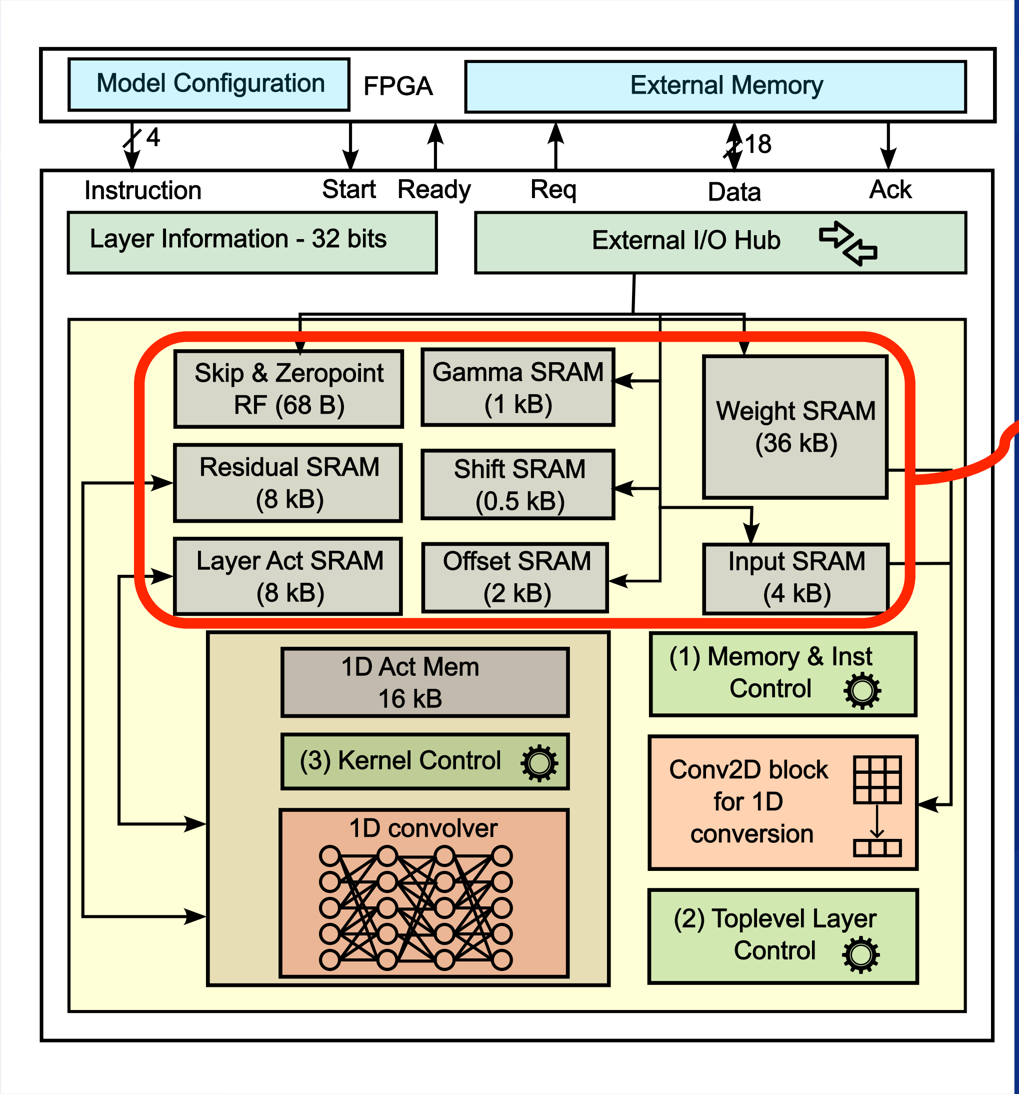
High Level Computation and Memory Blocks



- Reconfigurable Chip Architecture:
  - Dynamic configuration for 2D/1D convolution operations, tailored to input and kernel requirements.

Top-level Chip Architecture

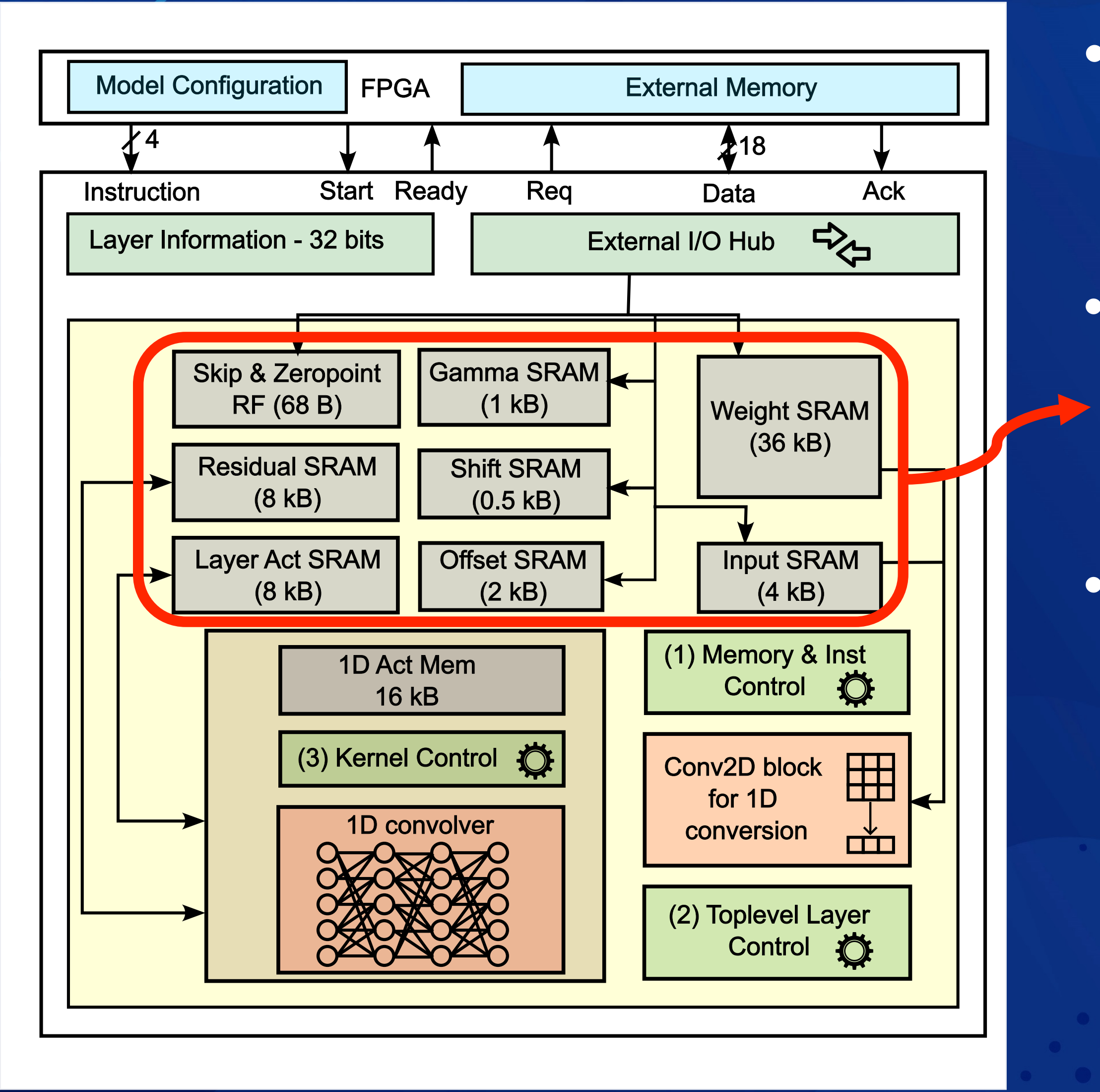
High Level Computation and Memory Blocks



- Reconfigurable Chip Architecture:
  - Dynamic configuration for 2D/1D convolution operations, tailored to input and kernel requirements.
- On-Chip Data Loading:
  - All weights and quantization parameters preloaded, minimizing external data transfers to reduce latency and power.

Top-level Chip Architecture

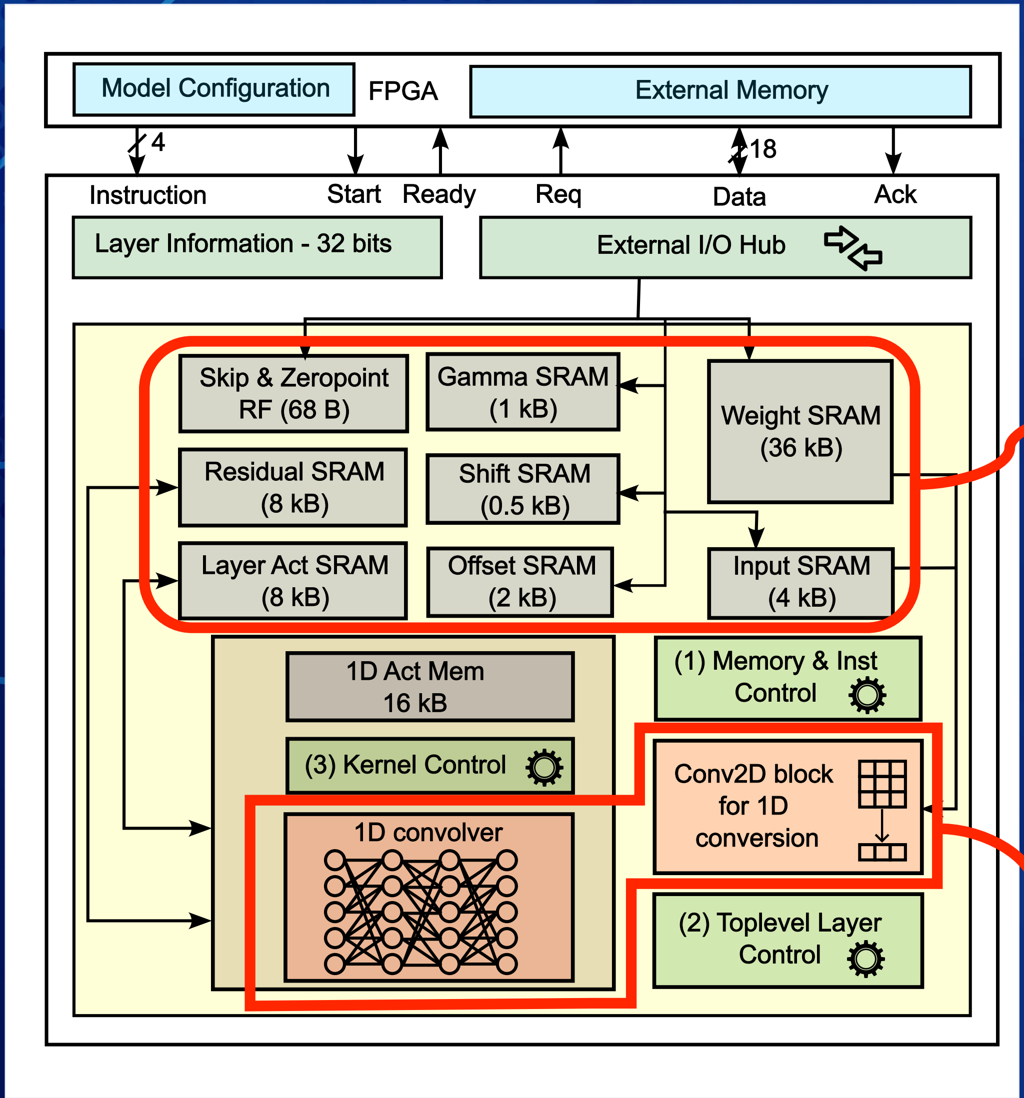
High Level Computation and Memory Blocks



- Reconfigurable Chip Architecture:
  - Dynamic configuration for 2D/1D convolution operations, tailored to input and kernel requirements.
- On-Chip Data Loading:
  - All weights and quantization parameters preloaded, minimizing external data transfers to reduce latency and power.
- Optimized Precision and Power:
  - Activations stored in 8-bit precision.
  - Power-gated memory logic reduces power during inactivity.

Top-level Chip Architecture

High Level Computation and Memory Blocks



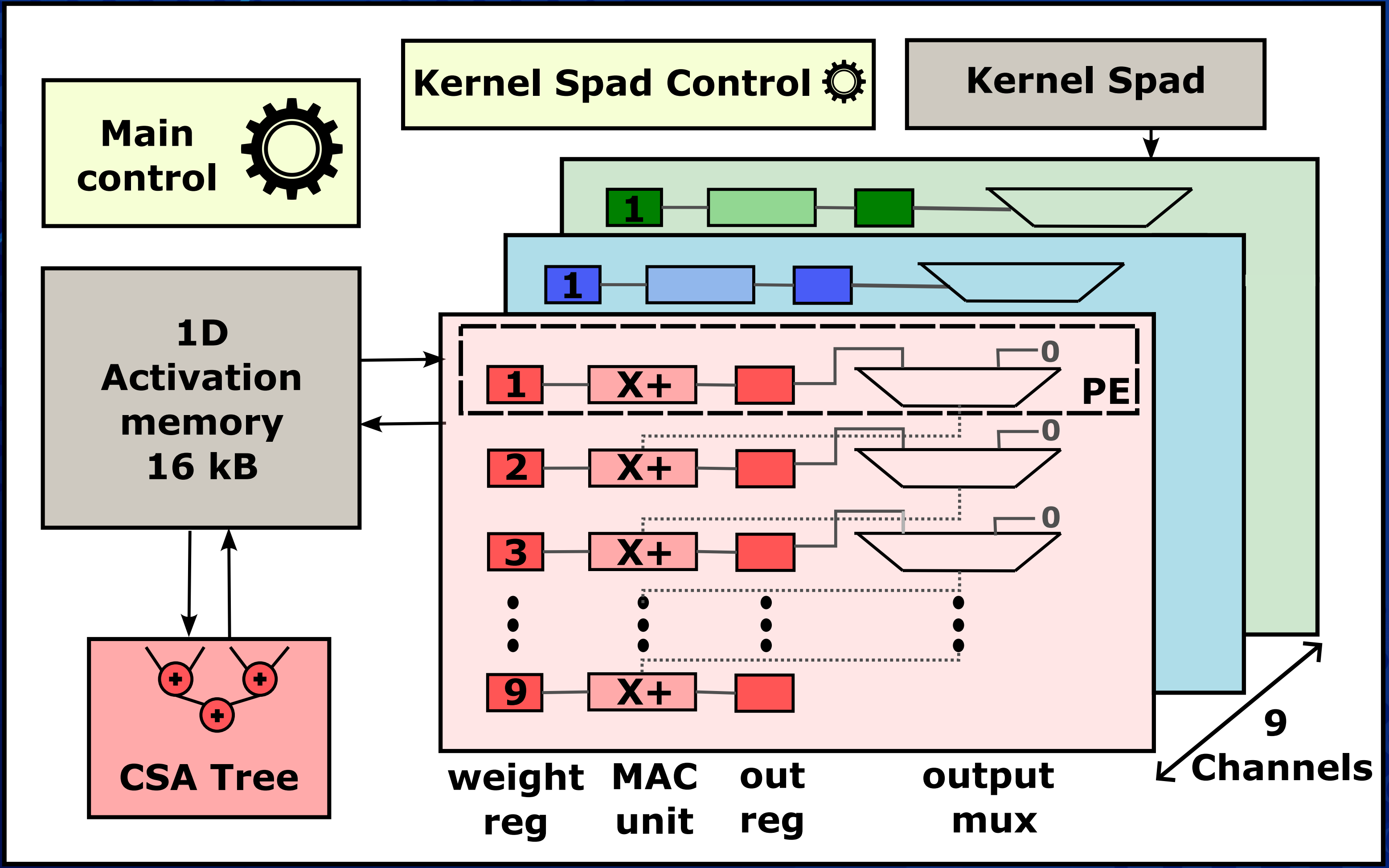
- Reconfigurable Chip Architecture:
  - Dynamic configuration for 2D/1D convolution operations, tailored to input and kernel requirements.
- On-Chip Data Loading:
  - All weights and quantization parameters preloaded, minimizing external data transfers to reduce latency and power.
- Optimized Precision and Power:
  - Activations stored in 8-bit precision.
  - Power-gated memory logic reduces power during inactivity.
- Efficient Data Processing:
  - Conv2D block reduces 2D to 1D; 1D convolver completes operations with residual memory for skip connections.

Outline

- Introduction
- **Design Features**
  - Algorithm Design
  - Quantization Scheme
  - Top-level Chip Architecture
  - **1D Convolution Dataflow**
- Results
- Conclusion

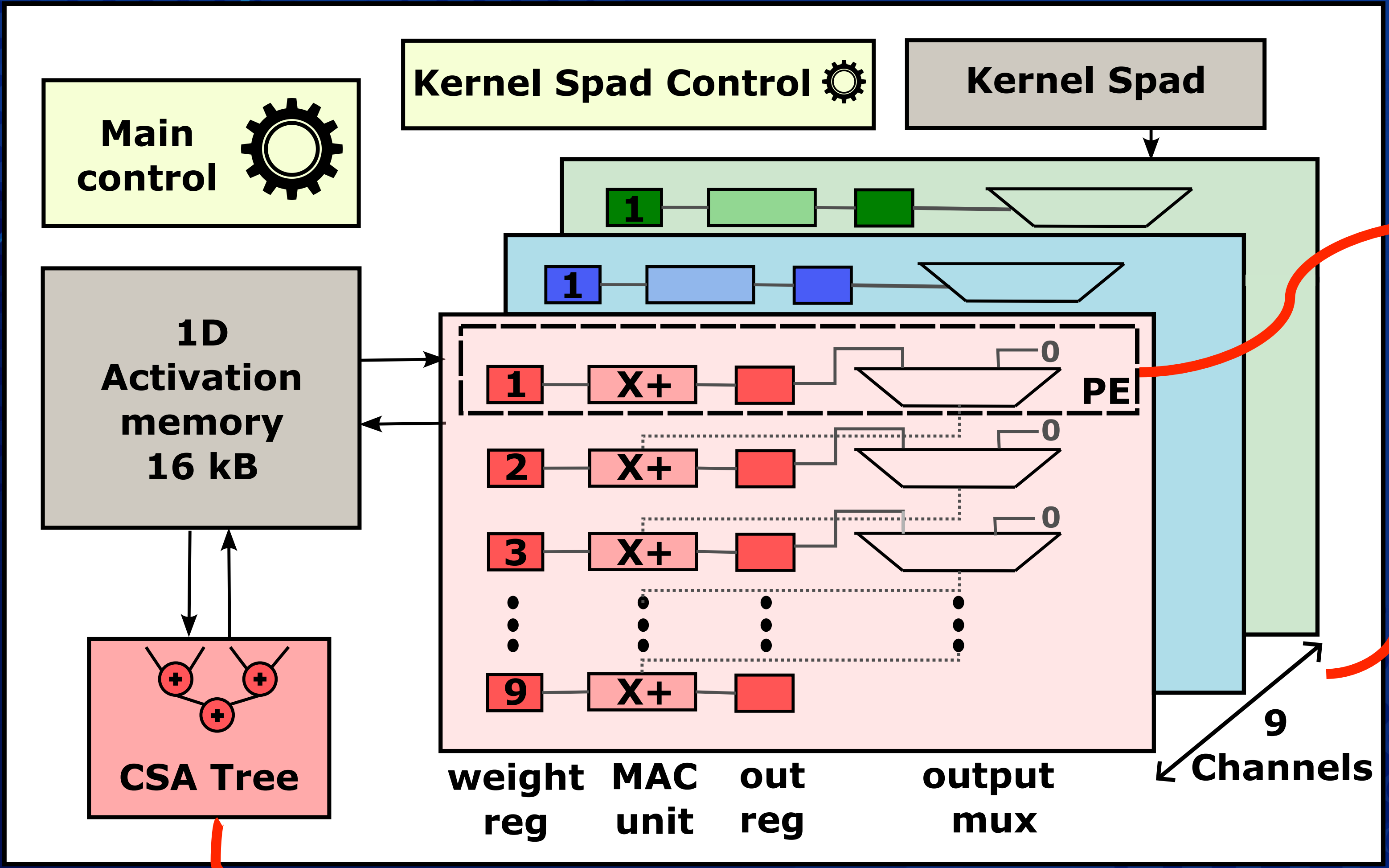
1D Convolution Dataflow

Architecture



1D Convolution Dataflow

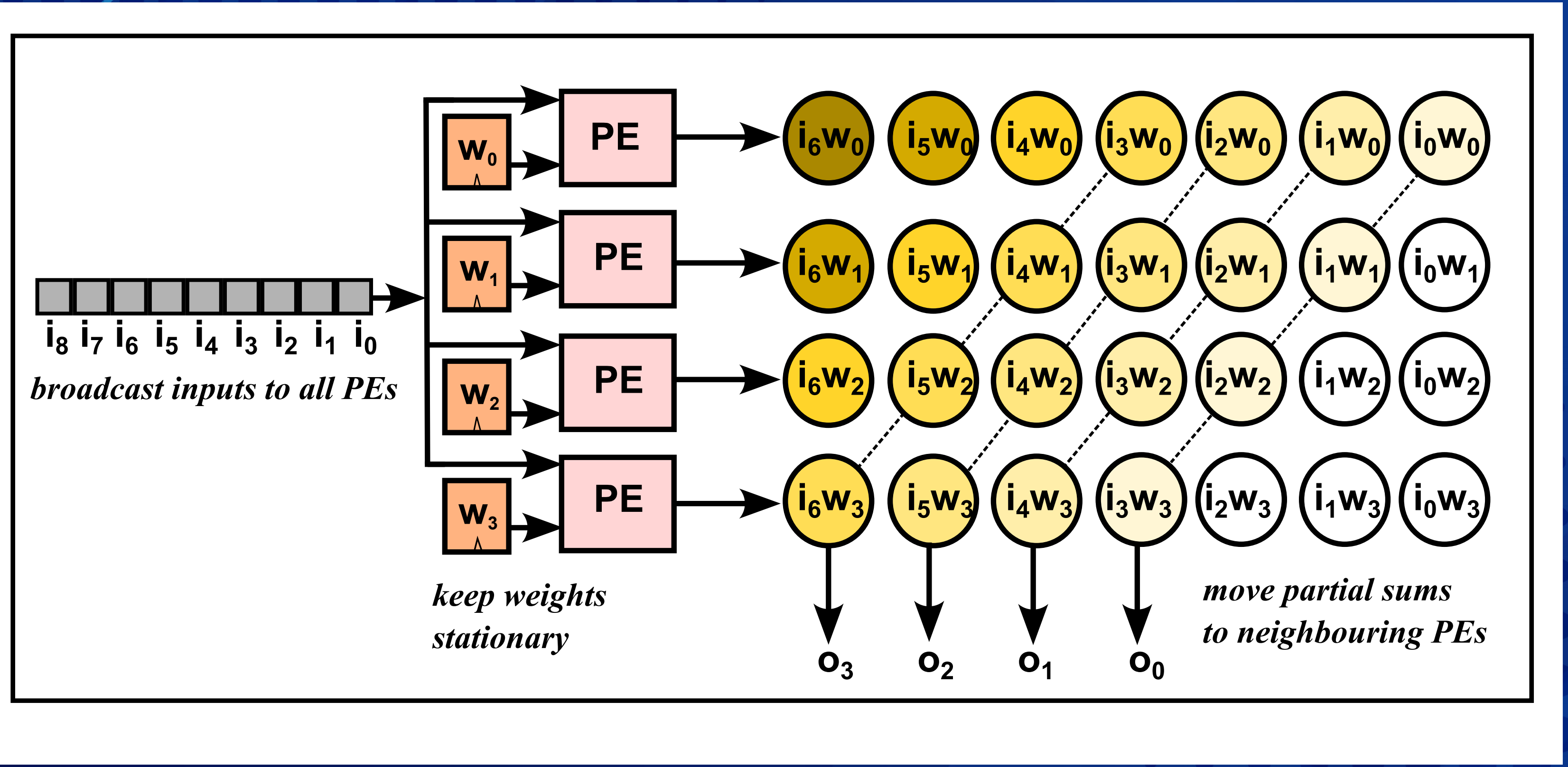
Architecture



- Processes one kernel at a time with PE
- Enables synchronous computation of up to 9 channels
- Carry-Save Adder Tree

1D Convolution Dataflow

PE Activation Routing Dataflow

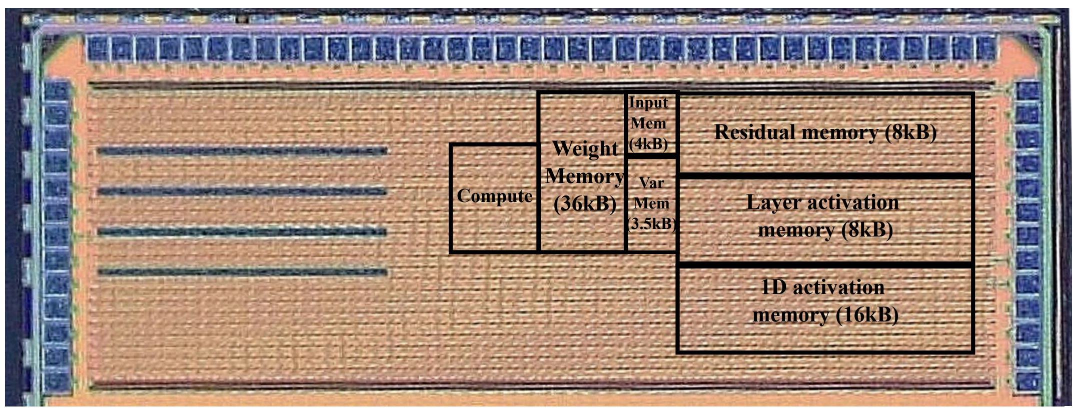


- Energy Optimization via Memory Access Reduction; PE input routing and weight mapping schemes
- Final PE computes a complete kernel convolution output every cycle

Outline

- Introduction
- Design Features
  - Algorithm Design
  - Quantization Scheme
  - Top-level Chip Architecture
  - 1D Convolution Dataflow
- **Results**
- Conclusion

Results

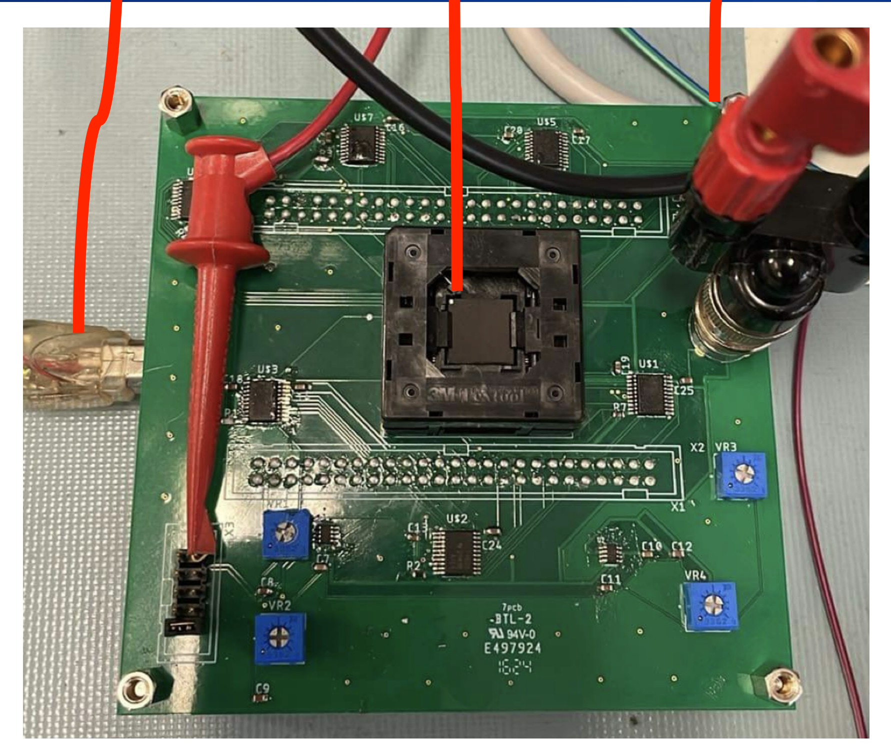


Technology	TSMC 28nm HPC+
Core area	0.75 mm^2
On-chip SRAM	75.5kB
Supply voltage	0.65 – 1V
Frequency	18.5MHz
Power	407μW (@ 0.65V, 18.5MHz)
Efficiency	3.24μJ/frame

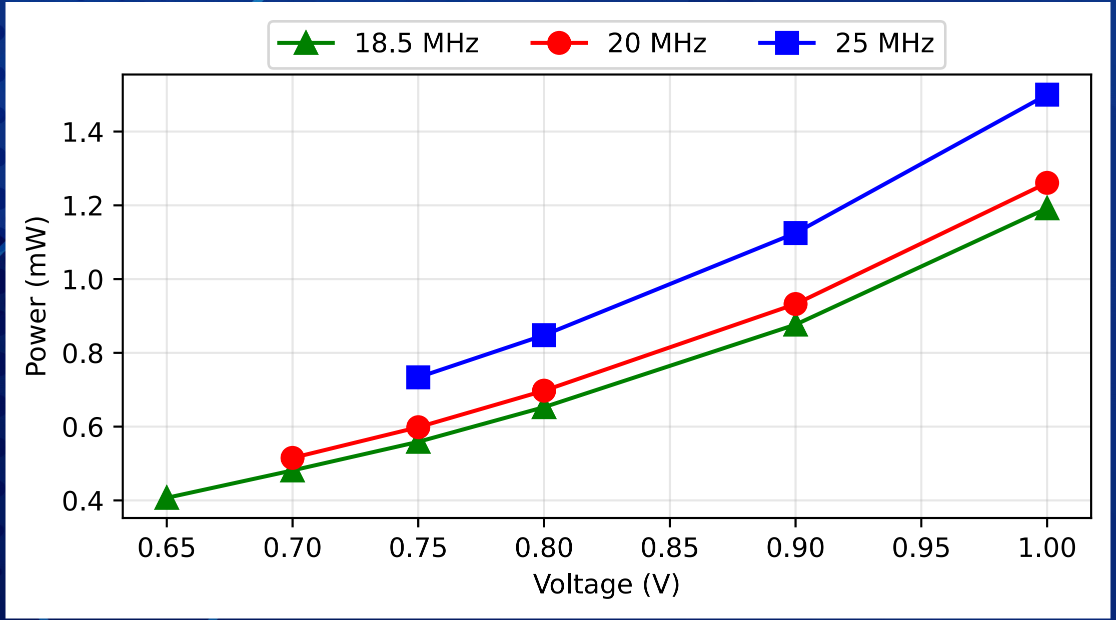
FPGA XEM7001 (below)

Packaged Die

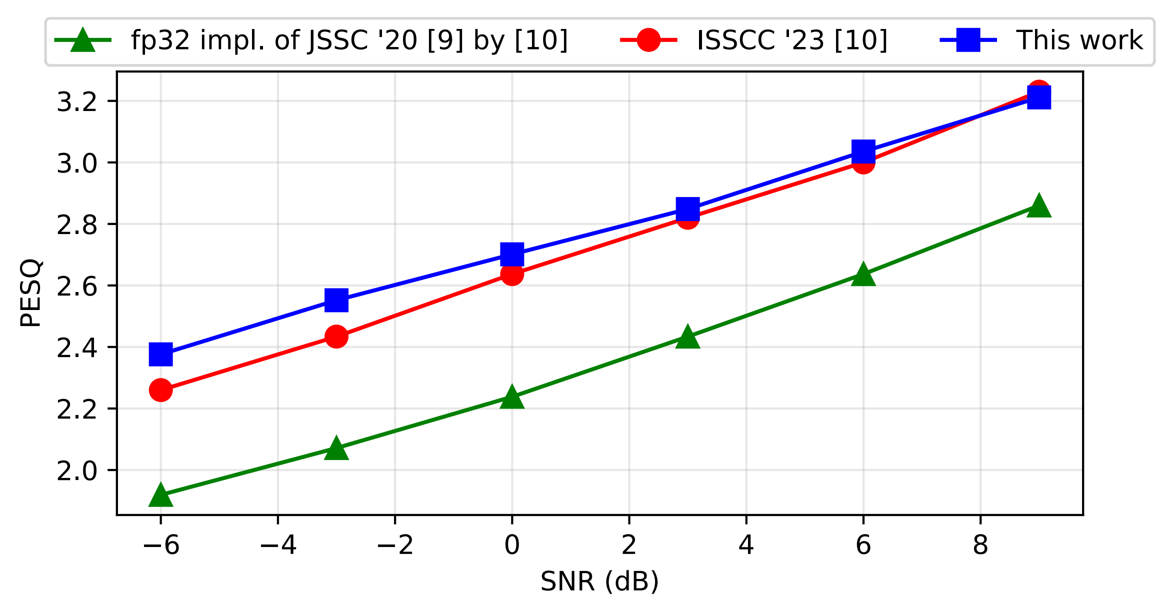
Measurement PCB



Results



Measured voltage scalability of this work



PESQ comparison with prior works

- [9] – CNN based FPGA design
  [10] - 1D depthwise-separable convolution layers, a gated recurrent unit based ASIC

[9] Y.C. Lee, T.S. Chi, C.H. Yang, IEEE JSSC, vol. 55, no. 8, pp. 275-282, Aug. 2020.
[10] S. Park, S. Lee, J. Park, H. S. Choi, D. Jeon, Proc. IEEE ISSCC, pp. 21-23, 2023.

Results

	TCAS-II'21 [7]	INTERSPEECH'20 [8]		JSSC'20 [9]	ISSCC'23 [10]	**This Work**
Implementation	Synthesized ASIC	FPGA	FPGA	ASIC	ASIC	**ASIC**
Technology (nm)	90	-	-	40	28	**28**
Core Area (mm^2)	11.4	-	-	4.2	0.81	**0.75**
FFT Window / Hop	512 / -	512 / 256	400 / 100	256 / 128	512 / 256	**256 / 64**
Frequency (MHz)	500	-	-	5 - 20	2.5 - 20	**18.5 – 25**
On-Chip SRAM (kB)		313.7	434.67	327	35	**75.5**
Power (mW)	636 (1.2V, 500MHz)	272	147.2	2.17 (0.6V, 5MHz)	0.74 (0.8V, 2.5MHz); 1.365 (1V, 2.5MHZ)[b]	**0.407[a] (0.65V, 18.5MHz)**
Frames/sec	-	63	160	125	63	**125**
Efficiency (µJ/frame)	10095.24[c]	4317.46	920	17.36	11.75	**3.24[a]**
Dataset	TIMIT	CHiME2	CHiME2	CHiME2	CHiME2	**CHiME2**
PESQ	1.52	-	-	2.38[d]	2.73	**2.79**

[a]excludes off-chip processing and STFT
[b]ML processing power from place-and-route netlist (excludes preprocessing: I/O Buffer, FFT, Window, Mel filter)
[c]As per [9], assuming hop size = 50% of FFT window size [d]fp32 implementation of [8] by [9]

[7] S. R. Chiluveru, et al., IEEE TCAS-II, vol. 68, no. 11, pp. 3461-3465, Nov. 2021.
[8] I. Fedorov et al., INTERSPEECH, pp. 4054-4058, 2020.
[9] Y.C. Lee, T.S. Chi, C.H. Yang, IEEE JSSC, vol. 55, no. 8, pp. 275-282, Aug. 2020.
[10] S. Park, S. Lee, J. Park, H. S. Choi, D. Jeon, Proc. IEEE ISSCC, pp. 21-23, 2023.

Results

	TCAS-II'21 [7]	INTERSPEECH'20 [8]		JSSC'20 [9]	ISSCC'23 [10]	**This Work**
Implementation	Synthesized ASIC	FPGA	FPGA	ASIC	ASIC	**ASIC**
Technology (nm)	90	-	-	40	28	**28**
Core Area (mm^2)	11.4	-	-	4.2	0.81	**0.75**
FFT Window / Hop	512 / -	512 / 256	400 / 100	256 / 128	512 / 256	**256 / 64**
Frequency (MHz)	500	-	-	5 - 20	2.5 - 20	**18.5 – 25**
On-Chip SRAM (kB)		313.7	434.67	327	35	**75.5**
Power (mW)	636 (1.2V, 500MHz)	272	147.2	2.17 (0.6V, 5MHz)	0.74 (0.8V, 2.5MHz); 1.365 (1V, 2.5MHZ)[b]	**0.407[a] (0.65V, 18.5MHz)**
Frames/sec	-	63	160	125	63	**125**
Efficiency (μJ/frame)	10095.24[c]	4317.46	920	17.36	11.75	**3.24[a]**
Dataset	TIMIT	CHiME2	CHiME2	CHiME2	CHiME2	**CHiME2**
PESQ	1.52	-	-	2.38[d]	2.73	**2.79**

[a]excludes off-chip processing and STFT
[b]ML processing power from place-and-route netlist (excludes preprocessing: I/O Buffer, FFT, Window, Mel filter)
[c]As per [9], assuming hop size = 50% of FFT window size [d]fp32 implementation of [8] by [9]

[7] S. R. Chiluveru, et al., IEEE TCAS-II, vol. 68, no. 11, pp. 3461-3465, Nov. 2021.
[8] I. Fedorov et al., INTERSPEECH, pp. 4054-4058, 2020.
[9] Y.C. Lee, T.S. Chi, C.H. Yang, IEEE JSSC, vol. 55, no. 8, pp. 275-282, Aug. 2020.
[10] S. Park, S. Lee, J. Park, H. S. Choi, D. Jeon, Proc. IEEE ISSCC, pp. 21-23, 2023.

Results

	TCAS-II'21 [7]	INTERSPEECH'20 [8]		JSSC'20 [9]	ISSCC'23 [10]	**This Work**
Implementation	Synthesized ASIC	FPGA	FPGA	ASIC	ASIC	**ASIC**
Technology (nm)	90	-	-	40	28	**28**
Core Area (mm^2)	11.4	-	-	4.2	0.81	**0.75**
FFT Window / Hop	512 / -	512 / 256	400 / 100	256 / 128	512 / 256	**256 / 64**
Frequency (MHz)	500	-	-	5 - 20	2.5 - 20	**18.5 – 25**
On-Chip SRAM (kB)		313.7	434.67	327	35	**75.5**
Power (mW)	636 (1.2V, 500MHz)	272	147.2	2.17 (0.6V, 5MHz)	0.74 (0.8V, 2.5MHz); 1.365 (1V, 2.5MHZ)[b]	**0.407[a] (0.65V, 18.5MHz)**
Frames/sec	-	63	160	125	63	**125**
Efficiency (μJ/frame)	10095.24[c]	4317.46	920	17.36	11.75	**3.24[a]**
Dataset	TIMIT	CHiME2	CHiME2	CHiME2	CHiME2	**CHiME2**
PESQ	1.52	-	-	2.38[d]	2.73	**2.79**

[a]excludes off-chip processing and STFT
[b]ML processing power from place-and-route netlist (excludes preprocessing: I/O Buffer, FFT, Window, Mel filter)
[c]As per [9], assuming hop size = 50% of FFT window size [d]fp32 implementation of [8] by [9]

[7] S. R. Chiluveru, et al., IEEE TCAS-II, vol. 68, no. 11, pp. 3461-3465, Nov. 2021.
[8] I. Fedorov et al., INTERSPEECH, pp. 4054-4058, 2020.
[9] Y.C. Lee, T.S. Chi, C.H. Yang, IEEE JSSC, vol. 55, no. 8, pp. 275-282, Aug. 2020.
[10] S. Park, S. Lee, J. Park, H. S. Choi, D. Jeon, Proc. IEEE ISSCC, pp. 21-23, 2023.

Results

	TCAS-II'21 [7]	INTERSPEECH'20 [8]		JSSC'20 [9]	ISSCC'23 [10]	**This Work**
Implementation	Synthesized ASIC	FPGA	FPGA	ASIC	ASIC	**ASIC**
Technology (nm)	90	-	-	40	28	**28**
Core Area (mm^2)	11.4	-	-	4.2	0.81	**0.75**
FFT Window / Hop	512 / -	512 / 256	400 / 100	256 / 128	512 / 256	**256 / 64**
Frequency (MHz)	500	-	-	5 - 20	2.5 - 20	**18.5 – 25**
On-Chip SRAM (kB)		313.7	434.67	327	35	**75.5**
Power (mW)	636 (1.2V, 500MHz)	272	147.2	2.17 (0.6V, 5MHz)	0.74 (0.8V, 2.5MHz); 1.365 (1V, 2.5MHZ)[b]	**0.407[a] (0.65V, 18.5MHz)**
Frames/sec	-	63	160	125	63	**125**
Efficiency (μJ/frame)	10095.24[c]	4317.46	920	17.36	11.75	**3.24[a]**
Dataset	TIMIT	CHiME2	CHiME2	CHiME2	CHiME2	**CHiME2**
PESQ	1.52	-	-	2.38[d]	2.73	**2.79**

[a]excludes off-chip processing and STFT
[b]ML processing power from place-and-route netlist (excludes preprocessing: I/O Buffer, FFT, Window, Mel filter)
[c]As per [9], assuming hop size = 50% of FFT window size [d]fp32 implementation of [8] by [9]

[7] S. R. Chiluveru, et al., IEEE TCAS-II, vol. 68, no. 11, pp. 3461-3465, Nov. 2021.
[8] I. Fedorov et al., INTERSPEECH, pp. 4054-4058, 2020.
[9] Y.C. Lee, T.S. Chi, C.H. Yang, IEEE JSSC, vol. 55, no. 8, pp. 275-282, Aug. 2020.
[10] S. Park, S. Lee, J. Park, H. S. Choi, D. Jeon, Proc. IEEE ISSCC, pp. 21-23, 2023.

Results

	TCAS-II'21 [7]	INTERSPEECH'20 [8]		JSSC'20 [9]	ISSCC'23 [10]	**This Work**
Implementation	Synthesized ASIC	FPGA	FPGA	ASIC	ASIC	**ASIC**
Technology (nm)	90	-	-	40	28	**28**
Core Area (mm^2)	11.4	-	-	4.2	0.81	**0.75**
FFT Window / Hop	512 / -	512 / 256	400 / 100	256 / 128	512 / 256	**256 / 64**
Frequency (MHz)	500	-	-	5 - 20	2.5 - 20	**18.5 – 25**
On-Chip SRAM (kB)		313.7	434.67	327	35	**75.5**
Power (mW)	636 (1.2V, 500MHz)	272	147.2	2.17 (0.6V, 5MHz)	0.74 (0.8V, 2.5MHz); 1.365 (1V, 2.5MHZ)[b]	**0.407[a] (0.65V, 18.5MHz)**
Frames/sec	-	63	160	125	63	**125**
Efficiency (μJ/frame)	10095.24[c]	4317.46	920	17.36	11.75	**3.24[a]**
Dataset	TIMIT	CHiME2	CHiME2	CHiME2	CHiME2	**CHiME2**
PESQ	1.52	-	-	2.38[d]	2.73	**2.79**

[a] excludes off-chip processing and STFT

[b] ML processing power from place-and-route netlist (excludes preprocessing: I/O Buffer, FFT, Window, Mel filter)

[c] As per [9], assuming hop size = 50% of FFT window size [d] fp32 implementation of [8] by [9]

[7] S. R. Chiluveru, et al., IEEE TCAS-II, vol. 68, no. 11, pp. 3461-3465, Nov. 2021.
[8] I. Fedorov et al., INTERSPEECH, pp. 4054-4058, 2020.
[9] Y.C. Lee, T.S. Chi, C.H. Yang, IEEE JSSC, vol. 55, no. 8, pp. 275-282, Aug. 2020.
[10] S. Park, S. Lee, J. Park, H. S. Choi, D. Jeon, Proc. IEEE ISSCC, pp. 21-23, 2023.

Outline

- Introduction
- Design Features
  - Algorithm Design
  - Quantization Scheme
  - Top-level Chip Architecture
  - 1D Convolution Dataflow
- Results
- **Conclusion**

Conclusion

Conclusion

- Quantized convolutional encoder-decoder model tailored for wearable IoT devices and hearing aids
- Hardware quantization to reduce memory and computational demands

Conclusion

- Quantized convolutional encoder-decoder model tailored for wearable IoT devices and hearing aids
- Hardware quantization to reduce memory and computational demands
- Low power (407µW), or 3.24µJ per frame at 0.65V and 18.5 MHz
- High audio quality (PESQ: highest among prior works)
- Real-time processing: < 8ms per frame at 18.5 MHz

Conclusion

- Quantized convolutional encoder-decoder model tailored for wearable IoT devices and hearing aids
- Hardware quantization to reduce memory and computational demands
- Low power (407µW), or 3.24µJ per frame at 0.65V and 18.5 MHz
- High audio quality (PESQ: highest among prior works)
- Real-time processing: < 8ms per frame at 18.5 MHz
- Future Work:
  - Integration of frequency transform computation with on-chip processor
  - Development of a complete system (including ADC and DAC)

Acknowledgment

- We would like to thank MIT-IBM Watson AI Lab for funding.
- We would also like to thank the TSMC University Shuttle Program for tapeout support.
- We would also like to thank Zexi Ji for the communication interface code.

Thank you!