SpikeGAN: An Energy-Efficient Spiking Generative Adversarial Network Design

Team Zynapse

Yuga Hanyu, Atharv Sharma and Aruki Komatsuzaki

School of Computer Science and Engineering
University of Aizu, Fukushima, Japan
E-mail: {m5291018, m5292015, s1310244}@u-aizu.ac.jp

Date: March 6, 2026

Spiking Neural Networks (SNNs)

- SNNs offer great energy efficiency.
- **Event-Driven processing**: computes only when there is a spike.
- **Addition-and-Accumulation**: computation consists of only addition and accumulation of spikes.

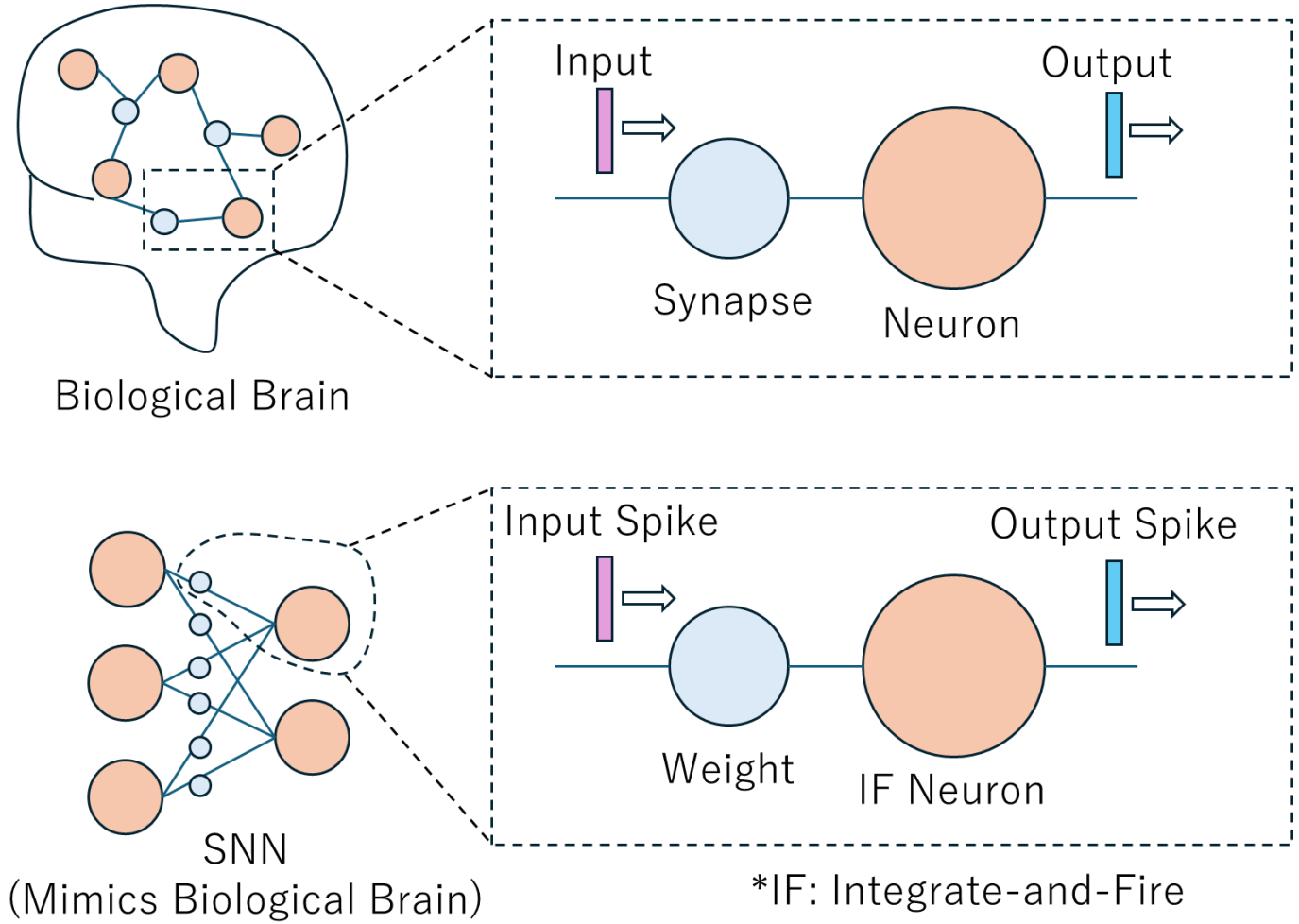


Fig. 1 Biological brain and Spiking Neural Network

Design Flow

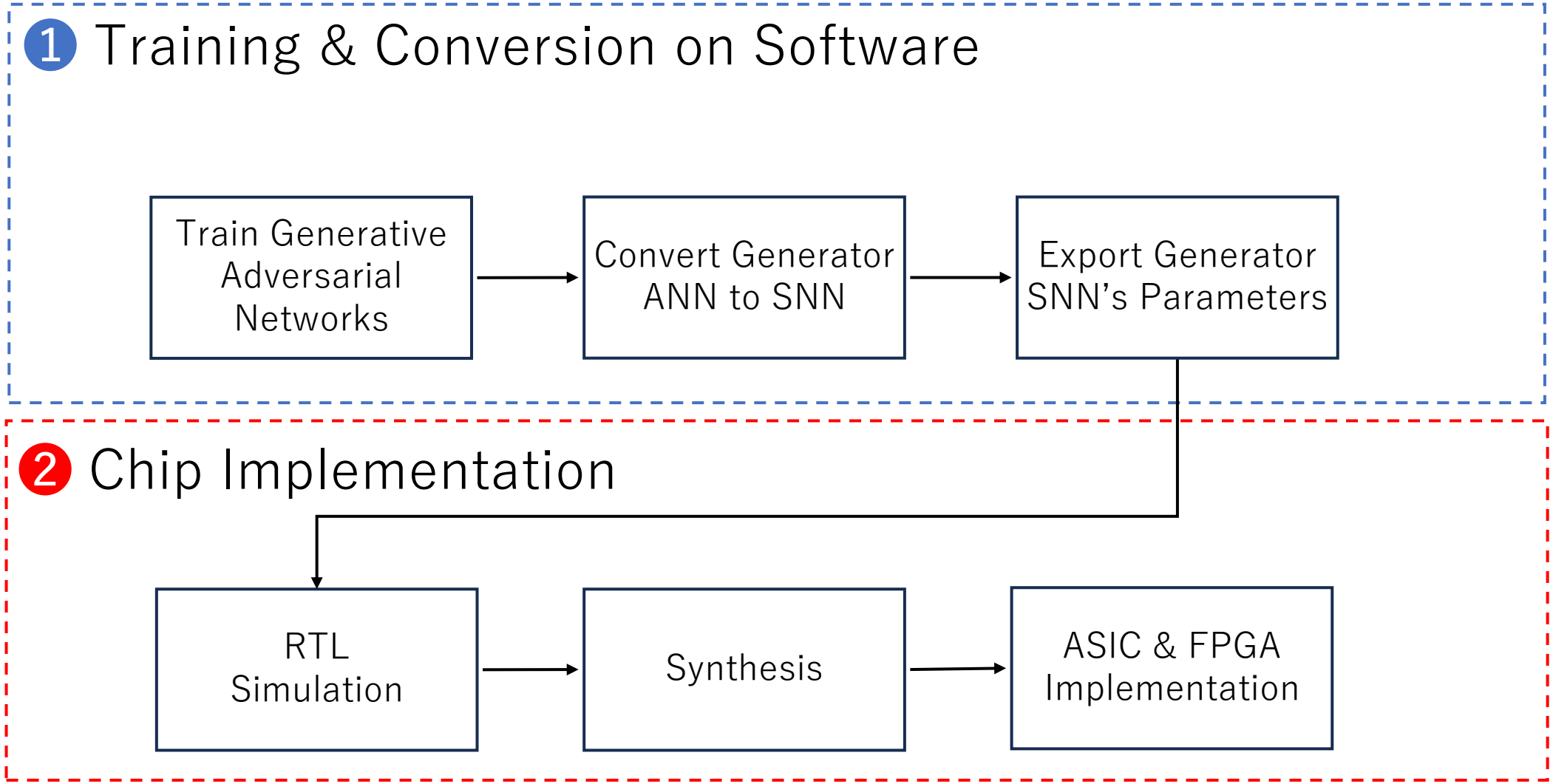


Fig. 2 Design flow of spiking implementation of GAN on Chip

Training and Conversion

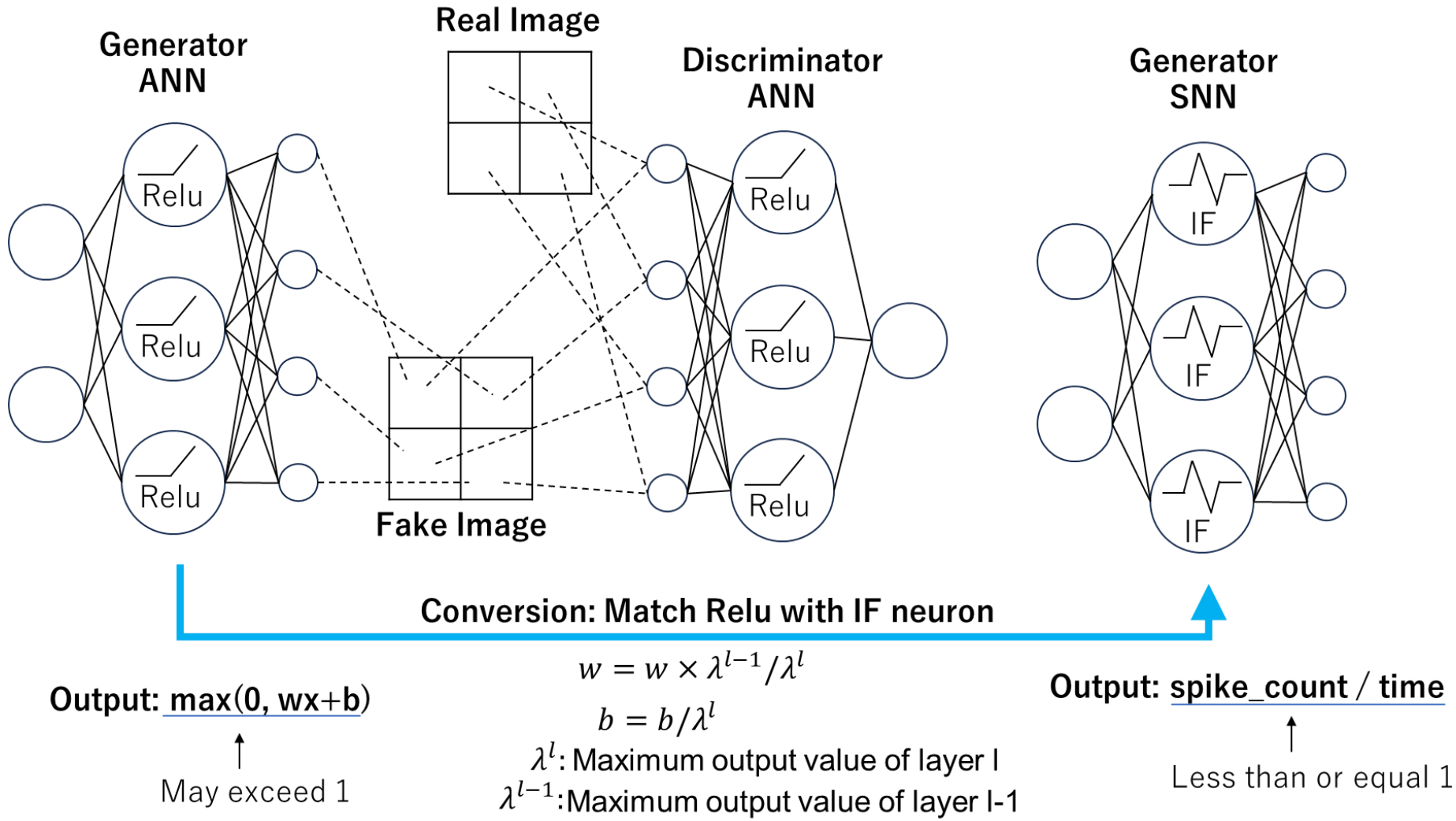


Fig. 3 Training of generator and discriminator ANNs and conversion to SNN

How to Obtain Image from SNN

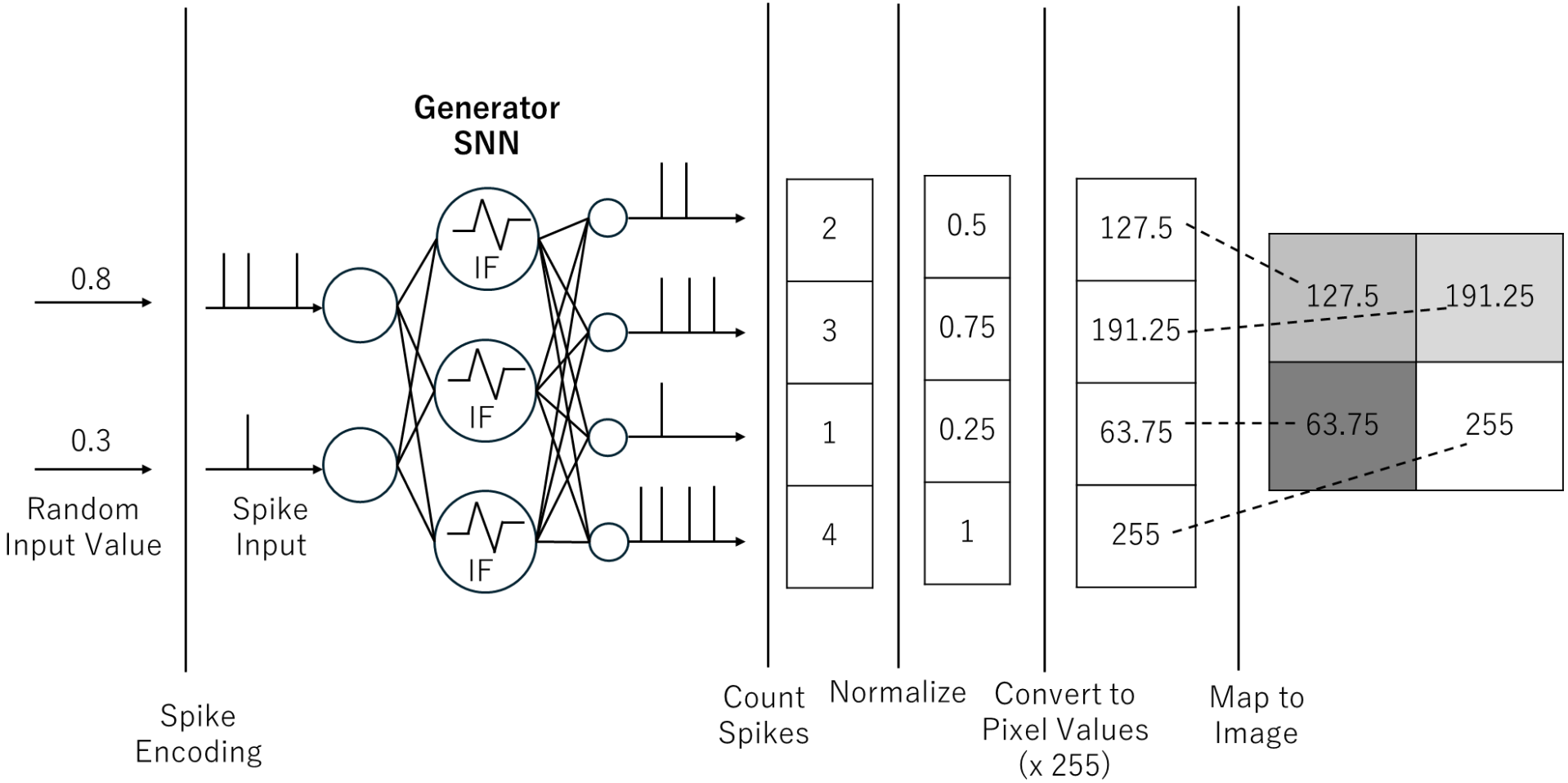


Fig. 4 Obtaining image from SNN

Architecture

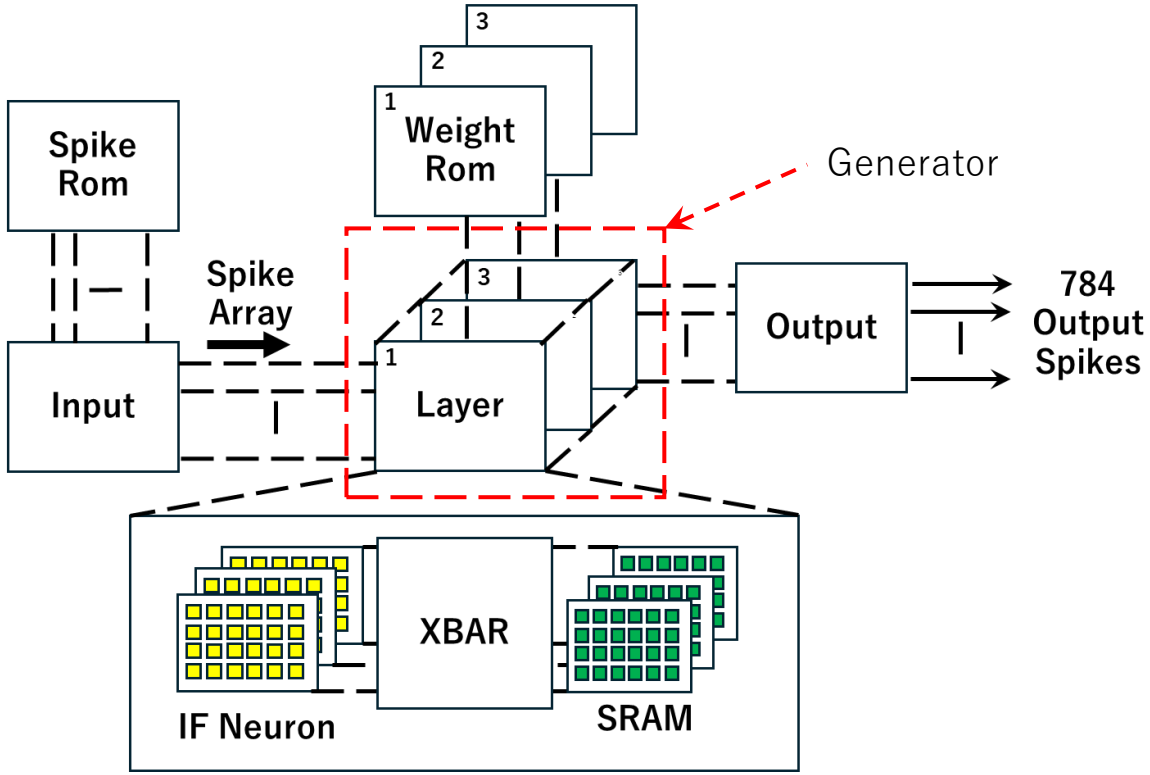


Fig. 5 Architecture of chip implementation

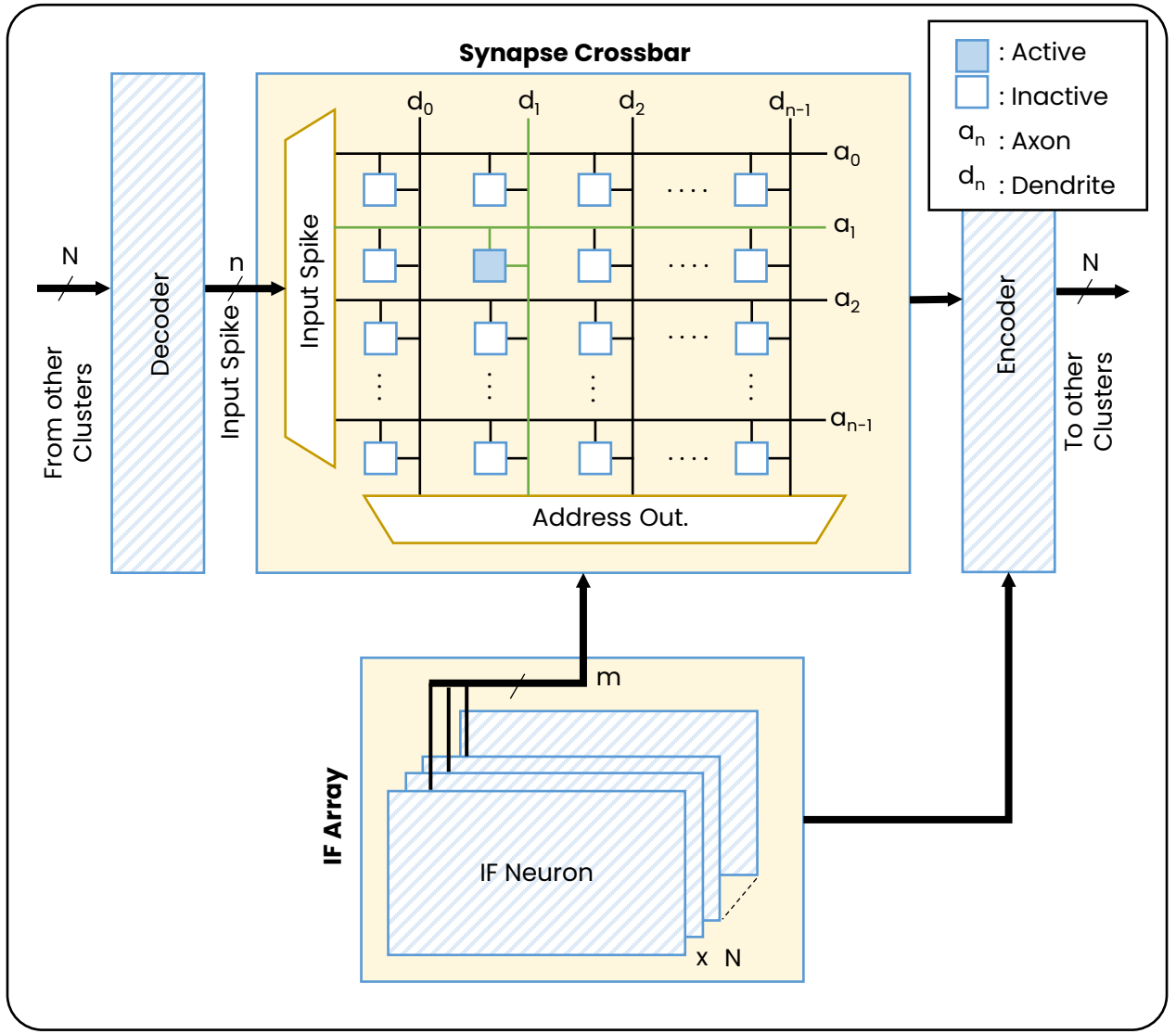
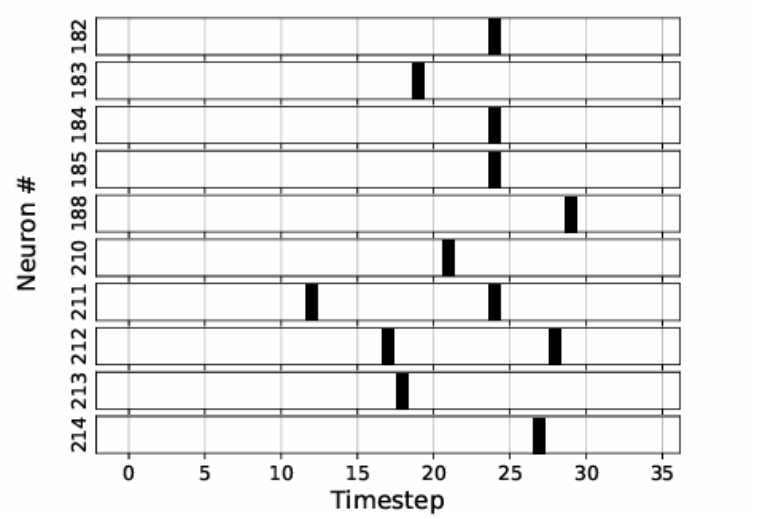


Fig. 6 Computing Core

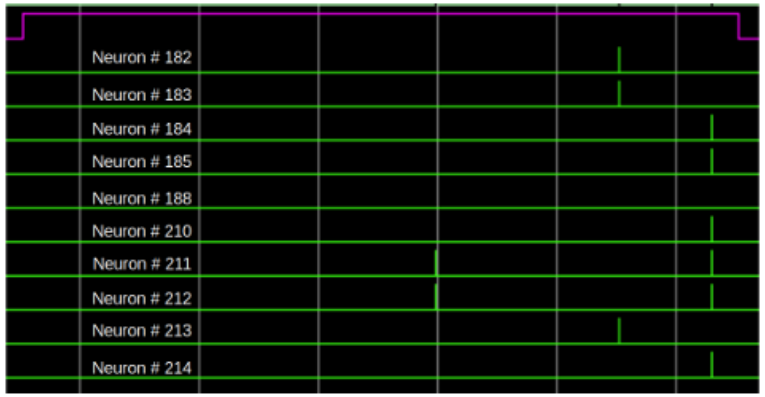
Evaluation

- Tools
  - GAN model training on software: Python, Pytorch library
  - ASIC implementation: Synopsys Design Compiler
  - FPGA implementation: Vivado 2019.1 and Artix-100T
  - Technology node: CMOS PDK 45nm
- Model Setup
  - Training data: MNIST Handwritten Digits
  - Training numbers: 600 images per batch x 15 epochs
  - Generator model size:
    - RTL Simulation: 100x1200x1200x784
    - ASIC and FPGA Implementation: 100x64x64x784
- Evaluation Methodology
  - Comparison between software and RTL simulation outputs
  - Implementation results in ASIC and FPGA

RTL Simulation

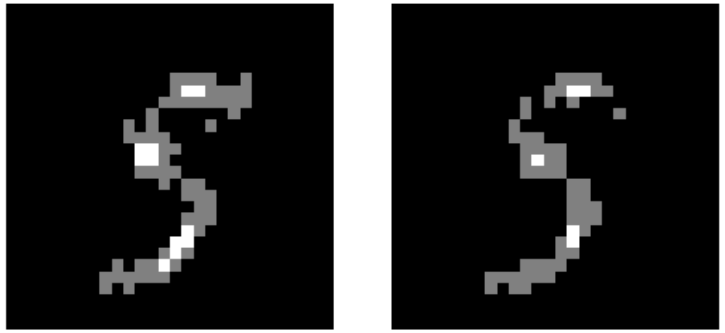


(a) Timing snapshot of software



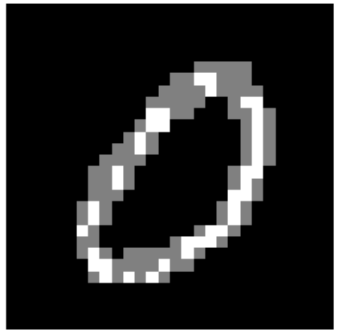
(b) Timing snapshot of RTL simulation with ModelSim

Fig. 7 Software and RTL Spike Timing Comparison

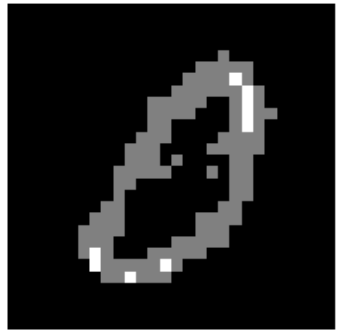


(a) Software output 1

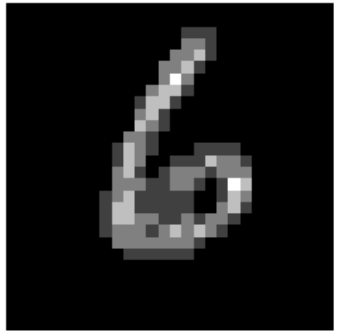
(b) RTL simulation output 1



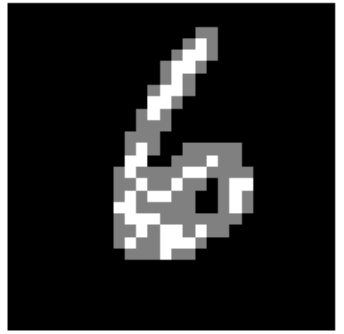
(c) Software output 2



(d) RTL simulation output 2



(e) Software output 3



(f) RTL simulation output 3

Fig. 8 Software and RTL Output Image Comparison

ASIC & FPGA Implementation

AREA SUMMARY OF ASIC SYNTHESIS

Category	Area
Combinational logic	151,198.92 μm^2
Sequential logic	71,773.18 μm^2
Memories	485,376 bits
Total	222,972.10 μm^2

RESOURCES UTILIZATION OF FPGA IMPLEMENTATION

Resource	Utilization	Available	Utilization (%)
LUT	46920	63400	74.01
LUTRAM	7296	19000	38.40
FF	29523	126800	23.28

Tools

- ASIC implementation: Synopsys Design Compiler
- FPGA implementation: Vivado 2019.1 and Artix-100T
- Technology node: CMOS PDK 45nm

Network structure: 100x64x64x784

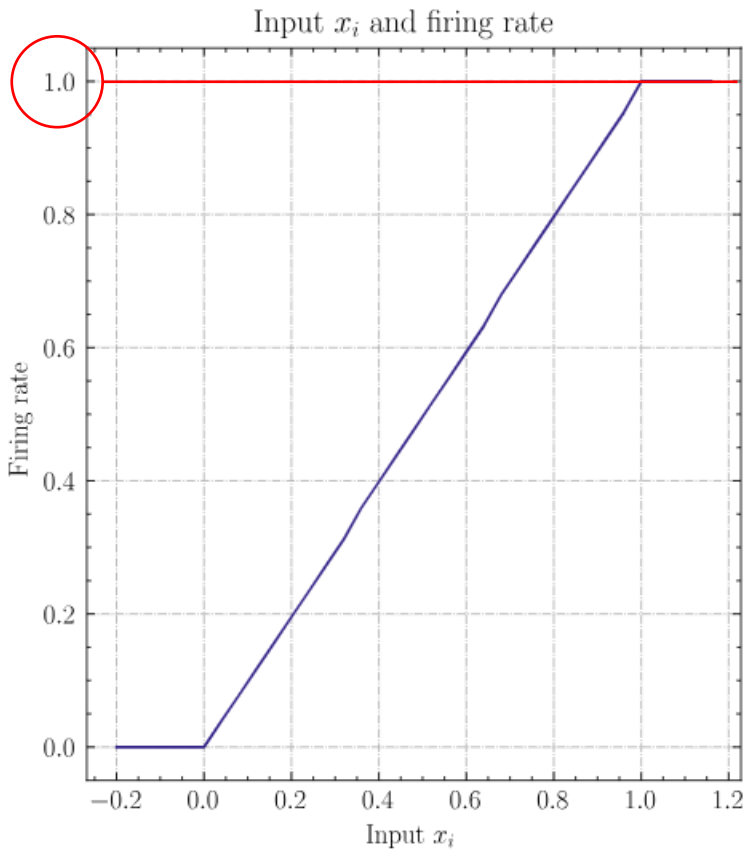
Conclusion

- ANN-based GAN was trained and generator was converted to the equivalent SNN.
- Synthesis & implementation were also successfully performed for ASIC and FPGA.
- Limitation: Due to memory constraint on-chip, model size needs to be reduced.
- Future work: Power consumption analysis for both ANN and SNN generator on chip & implement compression techniques.

Difference Between ReLu and IF Neuron

- Outputs from ReLu and IF neurons are similar.
- Output from IF neuron (firing rate) can not exceed 1.
- Weight normalization is necessary to match ReLu and IF neuron.

IF Neuron



ReLu