DATE: 2024/12/2

Department of Electrical and Electronic Information Engineering		ID	M213233
Name	Takayoshi Shikano		

Supervisor	Shuichi Ichikawa

Abstract

Title	Implementation of custom instructions in Rocket-chip: random number generation and stochastic computation

In fields such as image processing, machine learning, and security, which involve large-scale data processing, specialized hardware is widely used. The open-source instruction set architecture RISC-V allows users to define custom instructions, which can be utilized to reduce processing time. Moreover, in embedded applications with circuit resource constraints, unnecessary functions can be eliminated to reduce the circuit size.

This study aims to add custom instructions to the RISC-V compliant processor Rocket-chip and to evaluate their hardware implementation. Using the Rocket Custom Coprocessor (RoCC), two types of custom instructions were designed: (1) a random number generation instruction that retrieves the output of a Linear Feedback Shift Register (LFSR) with interlocking, and (2) a Gaussian filter instruction based on Stochastic Computing (SC).

For the random number generation instruction, the Unpredictable Random Number Generator (URNG) based on LFSR, proposed by Masaoka et al. (2021), was implemented. The evaluation metrics include the hardware resources utilization, the randomness quality, and the generation speed. The randomness quality was evaluated using the DIEHARD test and the NIST test.

The implementation of URNG on the Rocket-chip largely replicated the results of Kamogari and Ichikawa (2023). Additionally, the use of custom instructions eliminated the need for ad hoc adjustments that were required in Masaoka et al.'s (2021) method.

The Gaussian filter using SC was constructed with eight multiplexers (MUX) and Stochastic Numbers (SN). For image quality evaluation, monochrome (8-bit) SIDBA registered images were used as the input image with a resolution of 256×256 pixels. Three types of SN were investigated: an 8-bit counter ranging from 0 to 255, an 8-bit LFSR, and the lower 8 bits of a 32-bit LFSR. The quality of three Gaussian filters were compared: Binary Computing (BC), software SC written in C, and hardware SC using custom instructions. The Peak Signal-to-Noise Ratio (PSNR) was used to evaluate image quality, and the PSNR of SC images was compared against the BC image as a reference. Additionally, hardware resources and the number of cycles required for the Gaussian filter were investigated.

Among 13 SIDBA-registered images, 12 showed that the PSNR exceeded 35 dB for both the 8-bit LFSR and 32-bit LFSR in hardware SC when the SN bit length of each MUX was 32 bits. For the SN bit length of 32 bits, the hardware SC required 1.08 times larger instruction cycles compared to BC, whereas the software SC required 12.8 times more. For both types of custom instructions, the hardware resource overhead for the custom instructions remained below 1% compared to the original implementation.