DATE: 2021/1/8

Department of Electrical and Electronic Information Engineering		ID	M163265	Supervisor	Shuichi Ichikawa
Name	Kento Masanobu				

Abstract

Title	An investignaion of processor implementation methods by high level synthesis

In embedded systems, soft processors, which are written in hardware description language (HDL) and can be synthesized in FPGA, are widely used. In recent years, high-level synthesis (HLS) technology has matured to generate HDL descriptions automatically from behavioral descriptions in C and other languages. Skalicky et al. (2015) proposed a high-level synthesis of an instruction set simulator written in C to generate a soft processor. They also stated that the directives of the HLS tool can be used to generate the soft processor optimized for better throughput and performance. However, the effects of each directive has not been quantitatively evaluated.

Sakamoto (2018) and Iwamoto (2019) synthesized a soft processor from an instruction set simulator based on the method of Skalicky et al. and implemented application-specific instructions (special instructions) for the functions of each application of the CHStone benchmark. However, there are bugs in many of the implementations. The purpose of this study is to conduct replication on Iwamoto's work and to clarify the problems of his implementations. This work also verifies the differences in performance and circuit size of the soft processor by the directives of the HLS tool (Vivado HLS 2020.1).

The self-checking results of CHStone indicate that dfmul and dfsin without special instruction are defective, and the C simulation of jpeg aborted by errors. A typecasting error occurred in the description of the multiplication instruction for unsigned integers of the soft processor. By fixing the typecasting error, the calculation results for dfmul and dfsin matched. In the implementations with special instructions, only 15 implementations out of 39 presented the correct results.

In the implementation without special instruction, when the pragma *HLS array_partition complete*, which decomposes the array into individual elements, is specified in the 32 registers of the MIPS processor, the execution time (latency) is reduced by 17.0% on average, and the hardware resource usage (Slice number) increased by 477 %. In the implementation with special instruction, when the pragma *HLS unroll*, which automatically unroll loops, is specified in the loop description of the function of special instruction, the execution time is reduced by 5.8 % at maximum and the hardware resource usage is increased by 539 % at maximum. When the pragma *HLS pipeline*, which pipelines functions and loops, is specified in the function of special instruction, the execution time is reduced by 13.1 % at maximum and the hardware resource usage is increased by 267 % at maximum. If the specified directives are effective for optimization, there are trade-offs between the execution time and the resource usage.