DATE : 2016/01/08

Department of Electrical and Electronic Information Engineering		ID	M123251
Name	Tasuku Tanaka		

Supervisor	Shuichi Ichikawa
	Naoki Fujieda

Abstract

Title	Hardware implementation of PLC program with Vivado HLS

Control program of industrial machinery includes important technical information. Thus, techniques to protect it from analysis, plagiarism, and falsification are demanded. One of the techniques is a hardware implementation of control program, including the conversion of the control program into Hardware Description Language (HDL) for the implementation on Field Programmable Gate Array (FPGA). On the other hand, there are high-level synthesis tool such as Vivado HLS which converts general C or C++ program into HDL. This study examined some techniques of hardware implementation of program for Programmable Logic Controller (PLC) using Vivado HLS through an intermediate conversion into C program. In addition, this study examined a technique to reduce the latency and resource consumption by loop unrolling and dividing of loop.

Ichikawa et al. developed a tool to convert PLC program into hardware and presented Sequential Design (SD), Levelized Design (LD) and Flat Design (FD) as the design policies for the converted hardware. SD generates hardware that keeps the same order of control logic as the PLC program. LD generates hardware to multiple parts of control logic without dependency among them in parallel. FD generates hardware that implements the whole PLC program as a combinational circuit. In addition, they suggested a technique of resource constraint for SD and LD by limiting the number of functional units and share them among the parts of control logic.

In Vivado HLS, the circuit can be changed by rewriting a part of the program, adding pragma to the source code and changing synthesis options. This study examined the changes of the descriptions and options required to reproduce each of the design policies. This study used C program that was converted from PLC program using a tool that Ichikawa developed in collaborative research with a company. To reproduce SD, it was confirmed that no changes were required in the C program. In the reproduction of LD, this study examined two approaches. The first approach is pipelining of processing (LD-pipe). The second approach is converting control logic into function and applying the register renaming (LD-func). Although both approaches could reproduce LD, it was confirmed that LD-pipe was more effective than LD. FD could not be reproduced using Vivado HLS because intermediate results between the operations were unavoidably stored in the register. The resource constraint was realized by adding operations to config bind setting of Vivado HLS.

To reduce the number of execution cycles of the program, a loop unrolling, which is often used for acceleration of software, is also promising with Vivado HLS. However, loop unrolling might increase the logic scale at the same time. This study examines an approach to divide a large loop into several smaller loops to avoid the increase of logic scale by loop unrolling. Loop unrolling is performed by adding pragma (UNROLL directive) to the loop. When all loops of sample program is unrolled without dividing each loop, the number of cycles decreased by 36%, while resource consumption increased by 41%. AT product was improved by 11%. However, the logic scale after loop unrolling increased by too much division of the loop.