Project TRA: Data-Flow-Graphs Transformations

This project is a compulsory part of the examination for the Implementation of Digital Signal Processing course at the University of Twente. The goals of this project are:

- To apply the theory of data-flow-graph (DFG) transformations.
- To evaluate the effect of such transformations on the implementation of DFGs.
- To demonstrate the use of Arx in the verification of such implementations.

Preliminaries

This project is a continuation of Project MAP. The DFG to be used is the second-order IIR filter of that project. For the Arx exercises, the files provided for Project MAP should be used as a starting point including the values for the filter coefficients.

Exercise TRA-1: Pipelining and Retiming for Non-Overlapped Scheduling

Consider the second-order IIR filter of project MAP. The intention is to find the fastest possible *non-overlapped schedule* for the filter as in Exercise MAP-1. However, prior to scheduling, you are requested to apply a sequence of *pipelining and retiming* transformations to the DFG with the goal of *minimizing the critical path*. As before, you can assume that a multiplication requires two clock cycles and an addition one. Illustrate each step in the sequence by drawing the intermediate DFGs and the final one.

Next to the primary goal of minimizing the critical path, consider the reduction of hardware resources in the eventual implementation as a secondary optimization goal.

Comment on the effects of the transformation on the critical loop.

Exercise TRA-2: Data-Flow Verification

Verify the correctness of your solution for TRA-1 by creating a model in Arx and simulating it. Use the data-flow coding style as in file `sec_df2.arx` for this purpose.

When simulating with the same input stream as in MAP-1, do you expect any difference in the output? Motivate your answer and the check whether the actual output stream shows the behavior that you expect.

Exercise TRA-3: Scheduled Implemenation and Verification

Now, perform the scheduling of your solution and then fill all details on registers, multiplexers etc., as in MAP-1. Present as well a block diagram of the data path for this solution.

Then, check the correctness of this new design by running a simulation and showing that the produced output is exactly equal to the one of TRA-2 (when receiving the same input stream, of course). Include waveforms in your report from which it can be seen that multiplications have stable inputs during two consecutive clock cycles preceding the moment their outputs are used (simulate in C++ with VCD or in VHDL).

After having observed that the implementation has the expected output, synthesize the new design. Comment on the results comparing the new performance figures to the area and time numbers as reported in Exercise MAP-7.

Exercise TRA-4: Unfolding with a Factor 2

Unfold the original DFG from Project MAP (so not the modified one of TRA-1) by a factor of two. Do this in such a way that the topology of the original filter can be recognized in the unfolded graph as drawn by you. Assuming that a multiplication requires two clock cycles and an addition one, comment on the effects of the transformation on the critical path and the critical loop.

Exercise TRA-5: Data-Flow Verification of Unfolded Graph

Verify the correctness of your solution for TRA-4 by creating a model in Arx and simulating it. Use the data-flow coding style as in file `sec_df2.arx` for this purpose.

When simulating with the same input stream as in MAP-1, do you expect any difference in the output? Motivate your answer and the check whether the actual output stream shows the behavior that you expect.

Deliverables

Write a short report always motivating your choices and explaining the way you have reached your answers. Do not be verbose. In particular, do not copy the entire project description in your report. On the other hand, be as complete as possible providing details of your solutions by means of diagrams, (data-flow) graphs, tables, etc.

For Exercises involving Arx, supply the Arx code written by you, waveforms obtained from VHDL or C++ simulations (when applicable), Matlab plots, synthesis results, etc.

Grading

- TRA-1: 2 points
- TRA-2: 1 point
- TRA-3: 3 points
- TRA-4: 3 points
- TRA-5: 1 points

Go (back) to Sabih's Home Page.
Last update on: `Thu Mar 9 22:54:55 CET 2023` *by Sabih Gerez.*