2B－6

An Approach of Switching between Concurrence and Pipeline in Hardware Design

Xing-jian XU,　Mitsuru ISHIZUKA
The University of Tokyo

1. INTRODUCTION

In digital systems requiring high throughput rate, concurrence and/or pipelined designs are often used. The concurrence has a simple structure than pipeline; however, it requires more devices to implement and is limited to the case that input data is independent to each other. It is quite widely used in image processing, pattern recognition[1] and so on. Comparing with the concurrence processing, the pipeline has a complex structure, but the advantages of requirement of fewer devices and can be applied to dependent input data. In the high-level synthesis, the specification is written in a language like high-level programming language. The performance of a circuit is mainly depends on the iteration statements within its specification, the discussion in this paper is for the iteration statements.

2. CONCURRENCE AND PIPELINE

Selecting concurrence or pipeline is determined by the relationships between input and output data of operations. Two typical relationships between data are shown in figure 1. The input data of two operations in (a) are independent to each other, thus, these two operations can be processed at the same time, or concurrence processing can be applied to such kind of operations. The input data of two operation, and the second operation has to be processed after first operation, thus pipeline processing can be applied to such kind of operations instead of concurrence.

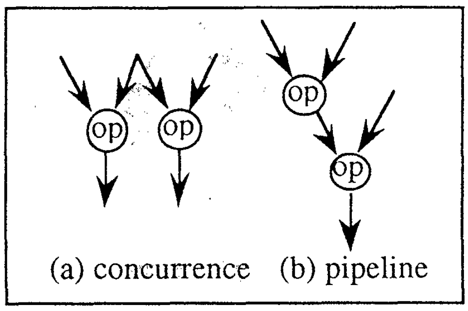

Figure 1 data relationships

With a supposition of that an iteration is executed for n times and it takes m clocks for a synthesized circuit to execute one loop, the execution times for each processing are:

non-parallel:	$n \cdot m$	(1)
concurrence:	$n \cdot k\ (k<m)$	(2)
pipeline:	$n + (m-1)$	(3)

Exp. 2 indicates that the processing time of m clocks is improved to k clocks by applying concurrence technology, where k<m. Exp. 3 is based on a supposition of that pipeline stage has 1 clock. As usual, we have exp. 4 listed bellow. By applying exp.4 to exp.3, exp. 5 can be obtained.

$$n \gg m \qquad (5)$$
$$n + (m-1) = n \qquad (6)$$

By comparing exp. 6 with exp. 2, it is clear that pipeline processing has a higher performance than concurrence processing for the same problem. Based on the results, the following stratagem is adopted:

ハードウィア設計におけるパイプラインと並列の切り替え法
徐　行倹、石塚　満
東京大学

apply pipeline processing to loop statements as possible, otherwise consider concurrence processing.

3. PIPELINE SYNTHESIS

Generally, there are many restrictions to synthesis a data-path from a loop statement. We are going to introduce one restriction and the method to remove it, which is shown in figure 2. In current loop, reading a variable operation, r1, is done q-1 clocks after the operation writing a new value to the variable, w1. Before r1 is finished, the write operation, w2, can not be executed to insure that r1 obtains a correct value set by w1. Thus the stage of the pipeline has to be q clocks at least, and its performance becomes n•q approximately. The advantages of pipeline processing could not be taken, especially when q is big enough.

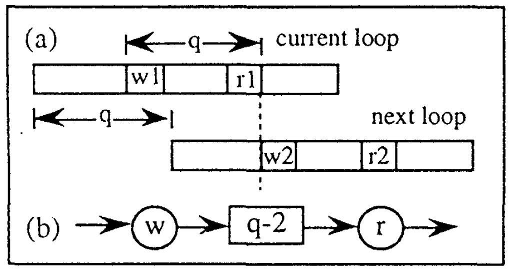

Figure 2 Read and write data within a loop

To improve the performance for that case, buffer with q-2 delays implemented by shift-registers as usual can be used to hold the value set by write operation w1 for the later use of read operation r1, which is shown in figure 2(b). This buffer can be implemented by connecting q-2 registers cascade the value saved in each register are propagated to next one and the last register inputs the value saved in it to read operation.

4. CONCURRENCE SYNTHESIS

For the loop statements, to which pipeline can not be applied, concurrent processing are considered. The methodology to synthesis a concurrent circuit is straightforward without any constraints of the area of circuit. The only thing has to be done is to find all the data-independent operations, allocate necessary functional units for them, and then schedule them into the same control step, or clock. Under the constraint of cost, all the necessary hardware can not be allocated to make possible operations into parallel. Thus these is a problem to make the best selection of hardware resource to achieve the highest processing speed. It becomes a problem of schedule of operation with a limited hardware resources, and there are some well-known approaches to this problem. There is a trade-off problem between processing speed and circuit area. We have proposed an approach to synthesis a data-path under the constraints of speed and area at the same time[3]. As there is not enough space here, the discussion about concurrent processing is omitted here.

5. CONCLUSION

In this paper, we introduced our stratagem to switch a circuit design between pipeline and concurrence architecture in high-level synthesis. And then a method to removed some restrictions in pipeline circuit synthesis were discussed.

REFERENCE

[1]. O. Hasegawa, W. Wongwarawiwat, C. W. Lee, M. Ishizuka: Real-time Moving Human Face Synthesis Using a Parallel Computer Network, IEEE IECON'91, pp. 1380-1385, Nov. 1991

[2]. X.J. Xu, M. Ishizuka: An Efficient Data-Path Synthesis Based on Algorithmic Description under the Constraints of Time and Area. Proc. of CHDL'93, pp119-pp126, April 1993