A Pseudo Delay-Insensitive Timing Model to Synthesising Low-Power Asynchronous Circuits

Oscar Garnica, Juan Lanchares, Roman Hermida
Universidad Complutense de Madrid. Dpto. Arquitectura de Computadores
ogarnica, julandan, rhermida@dacya.ucm.es

Abstract

The aim of this paper is to present a new approach to creating high performance, low-power and low-area asynchronous circuits using high level design tools. In order to achieve this, we introduce the new timing model on which this approach is based on. Following this, we present the results from comparing, for a set of benchmarks, our implementation with other implementations.

1 Pseudo Delay-Insensitive model

The pseudo delay-insensitive model (PDI model), as the DI model [1], does not suppose any upper bound in the delay of the circuit components (gates and wires). Consequently, it is necessary to use the two kinds of data proposed by Martin [1]: calculation data, *d*, and synchroniser data, *s*, which will be implemented using dual-rail logic. The chosen communication protocol is the four-cycle protocol. In this protocol, every computation is executed in two phases; a calculation phase followed by an synch phase.

The PDI model is a variation of the DI model, in which two suppositions are included which allow a reduction in the area required to face the drawbacks of the DI model.

Supposition 1. The critical path/paths are always the same. This implies that not all the circuit wires must be dual-rail wires. Hence, it is only necessary to transform into dual-rail logic those gates and wires within the critical path.

Supposition 2. The electrical signals propagate through the wire as waves. Hence, as a consequence, it is not necessary that the value, that we want to transmit, remains at the wire input until this value reaches the wire outputs.

In the synch phase the dual-rail wires are initialised to *s* data and the single-rail wires are initialised to the Boolean value, *d*, that does not determine the output of the dual-rail gates.

2 Experimental results

In this section we present the results for the well-known benchmarks LGSynth95. The design flow to build asynchronous circuits is: first, we construct the circuit using high level design tools; second, we determine the critical path using HLD tools; third, a precharged value is assigned to each single-rail wire using the criteria mentioned at the end of the previous section; fourth, all the gates and wires within the critical path are substituted by their dual-rail counterparts; and fifth, all the gates outside the critical path are substituted by Boolean gates with the capacity to pre-charge their outputs to the value calculated in item 3. The synchronous library used to synthesise the designs (using Design Compiler tool from Synopsys) only contains AND, OR and INV gates. In the asynchronous circuits, once the circuit has been synthesised, we have eliminated all the dual-rail INV gates, because inversion is simply to interchange the dual-rail wires assigned to the logic signals (as all complementary logics). It could be possible that after transforming the circuit in this way, the critical path changes. The solution adopted is to eliminate some INV gates but leave the number of INV gates which ensures that the critical path will remain as the critical path.

Our approach produces circuits with higher performance in all cases and lower power-consumption in 67% of the cases when compared with synchronous implementations. Similarly, it produces circuits with the same performance and lower power-consumption in 100% of the cases when they are compared with the DI implementations. What's more, this approach also produces circuits with lower area than its DI counterparts. Hence, the DI implementations use between 12% and 54% more transistors than PDI implementations. The synchronous implementations use between 13% and 27% less transistors than PDI implementations.

References

[1] S. Hauck. Asynchronous design methodologies: An overview. 83(1), Jan. 1995.